IMPLEMENTASI ALGORITME GRAIN V1 UNTUK ENKRIPSI GAMBAR PADA APLIKASI BERBASIS *WEB*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Andhika Brahmana Putra
NIM: 145150201111144

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

IMPLEMENTASI ALGORITME GRAIN V1 UNTUK ENKRIPSI GAMBAR PADA APLIKASI BERBASIS WEB

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Andhika Brahmana Putra
NIM: 145150201111144

Skripsi ini telah diuji dan dinyatakan lulus pada
3 Agustus 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Ari Kusyanti, S.T., M.Sc.	Mahendra Data, S.Kom., M.Kom
NIP: 19831228 201803 2 002	NIK: 201503 861117 1 001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 20 Juli 2018

Andhika Brahmana Putra

NIM: 145150201111144

KATA PENGANTAR

Assalamualaikum Wr. Wb. Alhamdulillah, puji syukur atas kehadirat Allah SWT atas limpahan rahmat dan hidayahnya sehingga skripsi ini dapat terselesaikan dengan baik. Skripsi ini disusun sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya.

Penulis menyadari bahwa tanpa bantuan, bimbingan, serta dorongan dari semua pihak, pengerjaan skripsi ini tidak mungkin bisa terwujud. Pada kesempatan ini penulis menyampaikan rasa terima kasih sebesar-besarnya kepada:

1. Ibu Sri Handayani dan Bapak Sumartono yang memberikan dukungan dan doa setiap saat.
2. Bapak Wayan Firdaus Mahmudy, S.Si, M.T, Ph.D selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya.
3. Bapak Tri Astoto Kurniawan , S.T, M.T, Ph.D selaku Ketua Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya.
4. Bapak Agus Wahyu Widodo , S.T, M.Cs selaku Ketua Program Studi Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya .
5. Ibu Ari Kusyanti, S.T, M.Sc selaku dosen pembimbing pertama yang telah memberikan bimbingan, saran, motivasi, dan pengarahan dalam penyusunan skripsi ini.
6. Bapak Mahendra Data, S.Kom., M.Kom selaku dosen pembimbing kedua yang telah memberikan bimbingan, saran, motivasi, dan pengarahan dalam penyusunan skripsi ini.
7. Bapak, Ibu dosen serta segenap staff dan karyawan Jurusan Teknik Informatika yang telah membantu dalam menyelesaikan skripsi ini.
8. Semua pihak yang telah memberikan bantuan baik secara langsung maupun tidak langsung dalam penyusunan skripsi ini.

Malang, 20 Juli 2018

Penulis

Andhikap120@gmail.com

ABSTRAK

Andhika Brahmana Putra, Implementasi Algoritme Grain V1 Untuk Enkripsi Gambar Pada Aplikasi Berbasis *Web*

Dosen Pembimbing: Ari Kusyanti S.T., M.Sc dan Mahendra Data S.Kom., M.Kom

Tantangan dalam menghadapi penyalinan ilegal dan distribusi dokumen multimedia menjadi suatu permasalahan yang harus diselesaikan untuk dapat melindungi sebuah data. Teknik yang berbeda telah diperkenalkan seperti enkripsi dan watermarking digital untuk dapat mengubah dokumen multimedia menggunakan algoritme agar tidak terbaca oleh siapapun kecuali pengguna yang sah. Pada penelitian ini dilakukan enkripsi data dalam format gambar melalui aplikasi berbasis *web* dengan media penyimpanan database menggunakan algoritme Grain. Algoritme Grain memiliki kelebihan yaitu algoritme yang berorientasi pada tiap bit bukan kata sehingga menurunkan nilai *complexity* dan meningkatkan kecepatan komputasi. Pengujian yang dilakukan yaitu pengujian validasi *test vector*, pengujian fungsionalitas, pengujian kinerja waktu dan pengujian keamanan. Untuk pengujian *test vector* dan fungsionalitas sistem yaitu dihasilkan nilai *valid*, untuk pengujian kinerja waktu pemrosesan enkripsi dan dekripsi pada setiap ekstensi file gambar dihasilkan waktu terbesar pada file ekstensi *JPG* sedangkan waktu terkecil pada file ekstensi *BMP* sedangkan untuk pengujian keamanan didapatkan hasil *sniffing* dengan menggunakan *wireshark* yaitu data sebelum di enkripsi masih dapat dibaca isi dari data tersebut dan data sesudah dienkripsi isi dari data tersebut sudah tidak dapat terbaca. Kesimpulan dari pengujian diatas ialah algoritme Grain dapat dilakukan dengan menerapkan mekanisme enkripsi gambar yang dapat melindungi kerahasiaan data agar tidak dapat terbaca oleh pihak yang tidak berwenang.

Kata kunci: aplikasi *web*, enkripsi, Grain, gambar

ABSTRACT

Andhika Brahmana Putra, Implementasi Algoritme Grain V1 Untuk Enkripsi Gambar Pada Aplikasi Berbasis *Web*

Supervisors: Ari Kusyanti S.T., M.Sc and Mahendra Data S.Kom., M.Kom

The challenge of dealing with illegal copying and distribution of multimedia documents is a matter to be solved in order to protect data. Different techniques have been introduced such as encryption and digital watermarking to be able to convert multimedia documents using algorithms to be unreadable by anyone except legitimate users. In this study, data encryption is carried out in image format through web-based applications with database storage media using the Grain algorithm. The Grain algorithm has the advantage of an algorithm that is oriented to each bit rather than a word so that it reduces the value of complexity and increases the speed of computing. Tests carried out are test vector validation testing, functionality testing, time performance testing and security testing. For test vector testing and system functionality that is generated valid values, for testing processing time performance encryption and decryption on each image file extension produced the largest time on JPG extension files while the smallest time on BMP extension files while for security testing obtained sniffing results using wireshark namely the data before being encrypted can still be read the contents of the data and the data after being encrypted the contents of the data cannot be read. The conclusion of the above test is that the Grain algorithm can be done by applying an Image encryption mechanism that can protect data confidentiality so that it cannot be read by unauthorized parties.

Keywords: *web application, encryption, Grain, image*

DAFTAR ISI

PENGESAHAN ii
PERNYATAAN ORISINALITAS iii
KATA PENGANTAR iv
ABSTRAK v
ABSTRACT vi
DAFTAR ISI vii
DAFTAR TABEL x
DAFTAR GAMBAR xi
DAFTAR LAMPIRAN xiii
BAB 1 PENDAHULUAN 1
1.1 Latar belakang 1
1.2 Rumusan masalah 2
1.3 Tujuan 2
1.4 Manfaat 2
1.5 Batasan masalah 2
1.6 Sistematika pembahasan 3
BAB 2 LANDASAN KEPUSTAKAAN 5
2.1 Kajian Pustaka 5
2.2 Dasar Teori 6
2.2.1 Kriptografi 6
2.2.2 Enkripsi dan Dekripsi 7
2.2.3 Algoritme Grain 7
2.2.4 *Image Processing* 10
2.2.5 *Web Based Application* 11
BAB 3 METODOLOGI 13
3.1 Analisis Kebutuhan 14
3.2 Perancangan Sistem 14
3.3 Implementasi 14
3.4 Pengujian 15
3.5 Analisa dan Pembahasan 15

3.6 Kesimpulan dan Saran 15
BAB 4 ANALISIS DAN PERANCANGAN 16
4.1 Analisis Kebutuhan 16
4.1.1 Analisis Permasalahan 16
4.1.2 Analisis Data 17
4.1.3 Analisis Keamanan Data 17
4.2 Perancangan Sistem 18
4.2.1 Flowchart Sistem 18
4.2.2 Desain Tampilan 19
4.3 Perancangan Algoritme 21
BAB 5 IMPLEMENTASI DAN PENGUJIAN 26
5.1 Implementasi 26
5.1.1 Algoritme Grain 26
5.1.2 Implementasi Pada Enkripsi Gambar 32
5.2 Parameter Pengujiam 34
5.3 Test Vector 34
5.3.1 Tujuan Pengujian 34
5.3.2 Prosedur Pengujian 34
5.3.3 Hasil Pengujian 34
5.4 Pengujian Fungsionalitas Sistem 35
5.4.1 Tujuan Pengujian 35
5.4.2 Prosedur Pengujian 35
5.4.3 Hasil Pengujian 35
5.5 Pengujian Kinerja Waktu Sistem 41
5.5.1 Tujuan Pengujian 41
5.5.2 Prosedur Pengujian 41
5.5.3 Hasil Pengujian 44
5.6 Pengujian Keamanan 49
5.6.1 Tujuan Pengujian 49
5.6.2 Prosedur Pengujian 49
5.6.3 Hasil Pengujian 49

BAB 6 PENUTUP 55
6.1 Kesimpulan..... 55
6.2 Saran 55
DAFTAR PUSTAKA..... 56
LAMPIRAN 57

DAFTAR TABEL

Tabel 2. 1 Tabel Penelitian Sebelumnya 5
Tabel 2. 2 Tabel Kebenaran XOR 8
Tabel 4. 1 *hx* 21
Tabel 4. 2 *zi* 22
Tabel 4. 3 *fx* 22
Tabel 4. 4 *gx* 22
Tabel 4. 5 *hx* 24
Tabel 4. 6 *zi* 24
Tabel 4. 7 *fx* 24
Tabel 4. 8 *gx* 24
Tabel 5. 1 Tabel Spesifikasi Perangkat Lunak dan Perangkat Keras 26
Tabel 5. 2 Validitas *Test Vector* 35
Tabel 5. 3 Method *AddTask* (Validasi Input) 36
Tabel 5. 4 Inputan *Images* (Validasi Input) 37
Tabel 5. 5 Inputan *Key* (Validasi Input) 37
Tabel 5. 6 Inputan *IV* (Validasi Input) 38
Tabel 5. 7 Inputan *Task* (Validasi Input) 38
Tabel 5. 8 AddTask (Validasi Input) 39
Tabel 5. 9 Method *Search* 40
Tabel 5. 10 Method *Show Entries* 40
Tabel 5. 11 Kinerja Waktu Pemrosesan 44
Tabel 5. 12 Hasil Enkripsi dan Dekripsi 53

DAFTAR GAMBAR

Gambar 2. 1 Proses Enkripsi dan Dekripsi 7
Gambar 2. 2 Diagram Umum Grain *Cipher* 9
Gambar 2. 3 Inisialisasi Grain *Cipher* 10
Gambar 2. 4 Proses Perubahan Gambar 10
Gambar 3. 1 Diagram Alir Metodologi Penelitian 13
Gambar 4. 1 Event Diagram Enkripsi Gambar Pada Aplikasi Berbasis Web 16
Gambar 4. 2 Proses Kerja Algoritme Grain 17
Gambar 4. 3 Flowchart Enkripsi Gambar Pada Aplikasi Berbasis Web 18
Gambar 4. 4 Desain Tampilan Halaman Utama 19
Gambar 4. 5 Desain Tampilan Fitur Task 20
Gambar 4. 6 Inisialisasi Kunci 21
Gambar 4. 7 Pergeseran LFSR 23
Gambar 4. 8 Pergeseran NFSR 23
Gambar 4. 9 Pergeseran LFSR 25
Gambar 4. 10 Pergeseran NFSR 25
Gambar 5. 1 Validasi Fungsi *AddTask* 36
Gambar 5. 2 Hasil Validasi Fungsi *AddTask* 36
Gambar 5. 3 Validasi Fungsi *images* 37
Gambar 5. 4 Validasi Fungsi *key* 38
Gambar 5. 5 Validasi Fungsi Initial Value 38
Gambar 5. 6 Validasi Fungsi *Task* 39
Gambar 5. 7 Validasi Fungsi *Content AddTask* 39
Gambar 5. 8 Validasi Fungsi Proses 40
Gambar 5. 9 Validasi Fungsi *Search* 40
Gambar 5. 10 Validasi Fungsi *Show Entries* 41
Gambar 5. 11 Pengujian Kinerja Waktu Sample PNG 42
Gambar 5. 12 Pengujian Kinerja Waktu Sample JPG 42
Gambar 5. 13 Pengujian Kinerja Waktu Sample GIF 43
Gambar 5. 14 Pengujian Kinerja Waktu Sample BMP 43
Gambar 5. 15 Perbandingan Kinerja Waktu Enkripsi 48

Gambar 5. 16 Perbandingan Kinerja Waktu Enkripsi..... 48
Gambar 5. 17 Hasil Pencarian Data..... 50
Gambar 5. 18 Gambar Asli Contoh Sample1..... 51
Gambar 5. 19 Hasil Enkripsi dan Dekripsi 51
Gambar 5. 20 Gambar Asli Contoh Sample2..... 52
Gambar 5. 21 Hasil Enkripsi dan Dekripsi 53

DAFTAR LAMPIRAN

Lampiran A. 1 Source Code Inisialisasi Key 57
Lampiran A. 2 Source Code Add Task 61
Lampiran A. 3 Source Code Fetch 63
Lampiran A. 4 Source Code Delete 65
Lampiran A. 5 Source Code Job..... 66
Lampiran A. 6 Source Code Custom.js 69

BAB 1 PENDAHULUAN

1.1 Latar belakang

Pada akhir abad ke-20, ditandai dengan adanya revolusi teknis yang luar biasa dari analog ke numerik karena dokumen dan peralatan semakin banyak digunakan di berbagai domain. Namun, keuntungan revolusi digital tidak tercapai tanpa kekurangan seperti penyalinan ilegal dan distribusi dokumen multimedia digital. Untuk memenuhi tantangan ini, para peneliti lebih termotivasi untuk melindungi dokumen multimedia dengan teknik perlindungan dokumen baru dan efisien. Dalam konteks ini, teknik yang berbeda telah diperkenalkan seperti enkripsi dan *watermarking* digital. Yang pertama terdiri dalam mengubah dokumen multimedia menggunakan algoritme agar tidak terbaca oleh siapapun kecuali pengguna yang sah (Loukhaoukha, Chouinard , & Berdai, 2012).

Permasalahan untuk melindungi data multimedia tersebut telah diteliti oleh Roshni Padate dan Aamna Patel pada jurnalnya yang berjudul *Image Encryption And Decryption Using AES Algorithm*, yang membahas tentang mekanisme pengamanan data berupa Gambar dengan menggunakan algoritme kriptografi yaitu *Advanced Encryption Standard* (AES) dengan mempertimbangkan aspek *Authentication, integrity,* dan *privacy/confidentiality.* Selanjutnya penelitian yang dilakukan Sanjay Kumar dan Sandeep Srivastava pada jurnalnya yang berjudul *Image Encryption using Simplified Data Encryption Standard* (S-DES) yang membahas tentang enkripsi Gambar. Algoritme yang dipakai merupakan penyempurnaan dari algoritme *Data Encryption Standard* (DES). Pada beberapa jurnal tersebut memiliki permasalahan yaitu hanya dapat melakukan input gambar dengan 2 jenis warna yaitu hitam dan putih atau *gray scale* sehingga algoritme tersebut kurang cocok digunakan untuk multimedia baik audio maupun visual karena beberapa alasan seperti kutipan jurnal Institute of Electrical and Electronics Engineers berikut ini.

Banyak aplikasi Internet seperti konferensi video, album foto online pribadi dan televisi kabel memerlukan cara untuk melakukan enkripsi gambar dan penyimpanan yang cepat dan efisien. Karena pesatnya pertumbuhan aplikasi multimedia melalui Internet, isu melindungi citra yaitu kerahasiaan gambar telah menjadi perhatian utama. Sejumlah besar penelitian telah dilakukan di bidang enkripsi citra selama dekade terakhir. Algoritme kunci simetris dan asimetris yang modern untuk enkripsi teks seperti *Blowfish, Data Encryption Standard, Advanced Encryption Standard, RSA, dan Elliptic Curve Cryptography* masing-masing tidak sesuai untuk gambar karena kapasitasnya dalam jumlah yang besar, redundansi dan korelasi yang tinggi diantara piksel (Chandrasekaran & Jayaraman, 2015).

Kelemahan pada penelitian sebelumnya penulis perbaiki dengan mengimplementasikan algoritme Grain V1 yang merupakan algoritme berjenis stream cipher dimana stream chiper memiliki kelebihan relatif lebih cepat dibanding dengan blok cipher sehingga sangat baik untuk digunakan dalam

penelitian. Algoritme Grain merupakan algoritme berjenis stream cipher yang masih baru yang hingga saat ini titik kelemahannya belum ditemukan hanya saja kecuali jika dibandingkan dengan algoritme lain yang didesain untuk software dengan kecepatan tinggi, Grain masih dimungkinkan untuk digunakan dalam perangkat lunak aplikasi umum (Hell, Johansson, & Meier, 2006). Algoritme Grain memiliki kelebihan yaitu algoritme yang berorientasi pada tiap bit bukan kata sehingga menurunkan nilai complexity dan meningkatkan kecepatan komputasi.

Hal tersebut yang menjadi dasar pertimbangan untuk menerapkan algoritme Grain pada enkripsi gambar sebab algoritme Grain memiliki beberapa faktor yang menentukan apakah suatu algoritme kriptografi baik atau tidaknya yaitu dari segi kecepatan, keamanan, dan kesederhanaan.

1.2 Rumusan masalah

1. Bagaimana mengimplementasikan algoritme Grain untuk enkripsi pada file dengan format gambar?
2. Bagaimana hasil validasi *test vector* terhadap keystream sistem pada Grain V1?
3. Bagaimana kinerja waktu pemrosesan enkripsi dan dekripsi algoritme Grain V1 pada setiap ekstensi file gambar?

1.3 Tujuan

Tujuan yang ingin dicapai berdasarkan rumusan masalah yaitu

1. Melakukan mekanisme pengamanan data yang telah ter-enkripsi berupa data dalam format gambar dalam mekanisme enkripsi gambar dengan konsep bit per pixel.
2. Mengetahui hasil validasi *test vector* terhadap keystream sistem pada Grain V1.
3. Mengetahui kinerja waktu pemrosesan enkripsi dan dekripsi algoritme Grain V1 pada setiap ekstensi file gambar.

1.4 Manfaat

1. Pengguna akan mendapatkan keamanan dalam hal kerahasiaan data agar data pengguna tidak dapat dibaca oleh pihak yang tidak berwenang .
2. Meminimalisir tindakan-tindakan negatif dalam hal pemanfaatan data pribadi pengguna.
3. Memberikan sebuah cara untuk mengamankan data dalam format gambar menggunakan algoritme Grain v1.

1.5 Batasan masalah

1. Hanya membahas kerahasiaan data.
2. Tidak membahas tampilan pada sistem.

3. Sistem ini berjalan pada aplikasi berbasis *web* menggunakan *apache web server.*
4. Penelitian ini menggunakan algoritme Grain versi 1.

1.6 Sistematika pembahasan

Sistematika pembahasan ini ditulis oleh penulis untuk mempermudah pembaca memahami penelitian ini, berikut gambaran sistematika pembahasan secara umum:

BAB I Pendahuluan

Pada bab I pendahuluan merupakan isi dari penelitian secara umum yaitu meliputi latar belakang, rumusan masalah, tujuan, manfaat, batasan masalah, dan sistematika pembahasan yang terdapat pada penelitian implementasi algoritme Grain V1 untuk enkripsi gambar pada aplikasi berbasis *web*.

BAB II Landasan Kepustakaan

Pada bab II landasan kepustakaan merupakan penjelasan dasar teori pada penelitian sebelumnya yang menjadi acuan dan landasan yang digunakan pada penelitian ini yang terdapat pada penelitian implementasi algoritme Grain V1 untuk enkripsi gambar pada aplikasi berbasis *web*.

BAB III Metodologi Penelitian

Pada bab III metodologi penelitian merupakan penjelasan tentang proses penyusunan yang dilakukan penulis dalam pembuatan penulisan yang meliputi analisis kebutuhan sistem, perancangan sistem, implementasi, pengujian, analisa dan pembahasan serta kesimpulan dan saran yang terdapat pada penelitian implementasi algoritme Grain V1 untuk enkripsi gambar pada aplikasi berbasis *web*.

BAB IV Analisis dan Perancangan

Pada bab IV analisis dan perancangan pada penelitian ini menjelaskan tentang hal-hal apa saja yang dibutuhkan sistem dan perancangan yang digukanan sistem yang meliputi analisis permasalahan, analisis data, dan analisis keamanan data, dan perancangan sistem yang terdapat pada penelitian implementasi algoritme Grain V1 untuk enkripsi gambar pada aplikasi berbasis *web*.

BAB V Implementasi dan Pengujian

Pada bab V implementasi dan pengujian yaitu tahap dilakukannya penerapan terhadap algoritme Grain V1 untuk enkripsi Gambar pada aplikasi berbasis web dan pengujian sistem yang akan diujikan setelah tahap implementasi yang meliputi pengujian validasi *test vector*, pengujian fungsional, pengujian performa sistem, dan pengujian enkripsi dan dekripsi yang terdapat pada penelitian implementasi algoritme Grain V1 untuk enkripsi gambar pada aplikasi berbasis *web*.

BAB VI Penutup

Pada bab VI penutup berisi tentang kesimpulan dan saran berdasarkan rumusan masalah dan tujuan yang digunakan pada penelitian implementasi algoritme Grain V1 untuk enkripsi gambar pada aplikasi berbasis *web*.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Kajian Pustaka

Pada Tabel 2.1 akan dijelaskan penelitian sebelumnya sebagai perbandingan terhadap penelitian yang dilakukan penulis. Pada judul penelitian nomor 1 dapat disimpulkan bahwa algoritme tersebut hanya dapat melakukan inputan gambar dengan 2 warna yaitu hitam dan putih sedangkan pada judul penelitian nomor 2 dapat disimpulkan bahwa *key* yang dibangkitkan yaitu secara acak bukan masukan atau inputan yang dilakukan oleh user. Pada algoritme Grain, penulis telah memperbaiki kedua permasalahan tersebut dengan memproses gambar berwarna dan menyertakan key berdasarkan inputan dari user.

Tabel 2.1 Penelitian Sebelumnya

No	Judul Penelitian	Metodologi	Hasil	Kelemahan
1	Image Encryption And Decryption Using AES Algorithm	Dalam penelitian ini cara yang dilakukan oleh peneliti yaitu dengan mengganti algoritme yang sudah ada sebelumnya yaitu DES dengan AES pada mekanisme enkripsi file berupa Gambar	Hasil pada penelitian ini yaitu algoritme AES hanya dapat melakukan input gambar dengan 2 jenis warna yaitu hitam dan putih	Pada jurnal tersebut pengimplementasian keamanan menggunakan panjang key 56 bits ukuran kunci yang terlalu kecil sehingga dianggap tidak aman digunakan untuk banyak aplikasi dan juga tidak dapat mengenkripsi gambar berwarna.
2	Image Encryption using Simplified Data Encryption Standard(S-DES)	Dalam penelitian ini cara yang dilakukan oleh peneliti yaitu dengan mengganti algoritme yang sudah ada sebelumnya yaitu DES dengan S-DES dengan metode chatic map pada mekanisme enkripsi file berupa Gambar	Dengan menggunakan chaotic image dari gambar asli sebagai *key* membuat enkripsi ini lebih aman dan memberikan kecepatan operasi yang cepat.	Karena DES merupakan algoritme yang dikembangkan oleh AES sehingga memiliki kerentanan yang sama disamping itu pemanfaatan pembangkitan kunci secara chaos (kacau) ini juga dapat menjadi bumerang. Masalah penyimpanan kunci

Tabel 2.1 Penelitian Sebelumnya (Lanjutan)

No	Judul Penelitian	Metodologi	Hasil	Kelemahan
2	Image Encryption using Simplified Data Encryption Standard(S-DES)			yang notabene tidak dapat diprediksi menjadi mutlak diperlukan. Hal ini menjadi dilema tersendiri mengingat untuk menyampaikan suatu cipherteks ke tujuan, kita harus menyertakan kunci agar tujuan pesan kita dapat melakukan proses dekripsi pesan yang telah kita berikan sebelumnya

2.2 Dasar Teori

2.2.1 Kriptografi

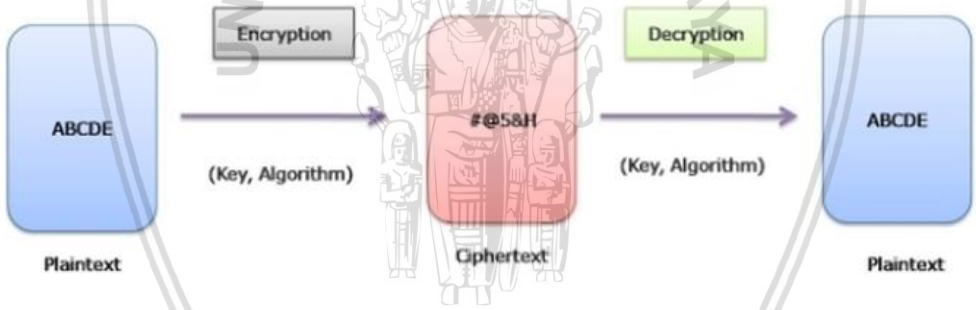
Kriptografi adalah ilmu mengenai teknik enkripsi dimana plaintext "naskah asli" diacak menggunakan suatu kunci enkripsi menjadi ciphertext "naskah acak yang sulit dibaca" oleh seseorang yang tidak memiliki kunci dekripsi. Dekripsi menggunakan kunci dekripsi bisa mendapatkan kembali data asli. Probabilitas mendapat kembali naskah asli oleh seseorang yang tidak mempunyai kunci dekripsi dalam waktu yang tidak terlalu lama adalah sangat kecil.Teknik enkripsi yang digunakan dalam kriptografi klasik adalah enkripsi simetris dimana kunci dekripsi sama dengan kunci enkripsi. Untuk *public key cryptography*, diperlukan teknik enkripsi asimetris dimana kunci dekripsi tidak sama dengan kunci enkripsi. Enkripsi, dekripsi dan pembuatan kunci untuk teknik enkripsi asimetris memerlukan komputasi yang lebih intensif dibandingkan enkripsi simetris, karena enkripsi asimetris menggunakan bilangan – bilangan yang sangat besar (Kromodimoeljo, 2010).

Kata kriptografi berasal dari bahasa Yunani, "kryptós" yang berarti tersembunyi dan "gráphein" yang berarti tulisan. Kriptografi telah digunakan oleh Julius Caesar sejak zaman Romawi Kuno. Teknik ini dijuluki Caesar cipher untuk mengirim pesan secara rahasia, meskipun teknik yang digunakannya sangat tidak memadai untuk ukuran kini. Casanova menggunakan pengetahuan mengenai kriptografi untuk mengelabui Madame d'Urfe (ia mengatakan kepada Madame d'Urfe bahwa sesosok jin memberi tahu kunci rahasia Madame d'Urfe

kepadanya, padahal ia berhasil memecahkan kunci rahasia berdasarkan pengetahuannya mengenai kriptografi), sehingga ia mampu mengontrol kehidupan Madame d'Urfe secara total (Kromodimoeljo, 2010).

2.2.2 Enkripsi dan Dekripsi

Secara garis besar, proses enkripsi adalah proses pengacakan "naskah asli" (plaintext) menjadi "naskah acak" (ciphertext) yang "sulit untuk dibaca" oleh seseorang yang tidak mempunyai kunci dekripsi. Yang dimaksud dengan "sulit untuk dibaca" disini adalah probabilitas mendapat kembali naskah asli oleh seseorang yang tidak mempunyai kunci dekripsi dalam waktu yang tidak terlalu lama adalah sangat kecil. Jadi suatu proses enkripsi yang baik menghasilkan naskah acak yang memerlukan waktu yang lama (contohnya satu juta tahun)1 untuk didekripsi oleh seseorang yang tidak mempunyai kunci dekripsi. Satu cara untuk mendapatkan kembali naskah asli tentunya dengan menerka kunci dekripsi, jadi proses menerka kunci dekripsi harus menjadi sesuatu yang sulit. Tentunya naskah acak harus dapat didekripsi oleh seseorang yang mempunyai kunci dekripsi untuk mendapatkan kembali naskah asli. Walaupun awalnya kriptografi digunakan untuk merahasiakan naskah teks, kini kriptografi digunakan untuk data apa saja yang berbentuk digital.



Gambar 2.1 Proses Enkripsi dan Dekripsi

2.2.3 Algoritme Grain

Grain Chiper pertama kali dibuat oleh Martin Hell, dkk pada bulan mei 2005. Grain dibagi menjadi 2, yaitu Grain v1 dan Grain-128. Sedangkan versi awal Grain (Grain v0) telah di-propose, namun tidak dipublish. Grain v1 memiliki 80 bit stream cipher dan 160 siklus serta menerima 64 bit IV sedangkan Grain-128 memiliki 128 bit stream cipher dan 256 siklus serta menerima 96 bit IV. Grain Chiper adalah *bit oriented synchronous stream chiper*. Pada *synchronous stream chiper,* keystream dibangkitkan secara terpisah dari plaintext. Grain chiper didasarkan pada dua shift register yaitu *linear feedback* (LFSR) dan *nonlinear feedback* (NFSR). Pada LFSR menjamin periode minimum untuk keystream dan juga menyediakan keseimbangan pada keluaran. Pada NFSR bersama dengan filter *nonliner* menyebabkan ketidaklinieran pada chiper. Masukan pada NFSR ditutupi dengan keluaran dari LFSR sehingga kondisi dari NFSR menjadi seimbang

dimana kedua shift register memiliki masing-masing 80 bit dan panjang IV (*initial value*) adalah 64 bit.

2.2.3.1 Cara Kerja Grain *Chiper*

Grain Chiper terdiri dari 3 bagian utama yaitu sebuah LFSR, sebuah NFSR, dan sebuah fungsi penyaring (*filter*). LFSR yaitu shift register yang bit masukannya merupakan fungsi linear dari state sebelumnya. Satu-satunya fungsi linear pada bit satuan adalah *exclusive-or* (xor), oleh karena itu LFSR adalah shift register yang bit masukannya dibangkitkan oleh *exclusive-or* (xor) dari beberapa bit dari keseluruhan nilai shift register.

Tabel 2.2 Tabel Kebenaran XOR

A	B	A XOR B
F	F	F
F	T	T
T	F	T
T	T	F

Pada Gambar 2.2 adalah bagian utama dari Grain yang terdiri dari sebuah LFSR, NFSR, dan *filter* (h(x)). Pada Grain v1 menggunakan key 80 bit, *initial value (IV)* 64, dan *internal state* 160 bit. Pada Grain, isi dari LFSR dinotasikan sebaga

$$s_i, s_{i+1}, \dots, s_{i+79} \tag{2.1}$$

Pada persamaan 2.2 fungsi umpan balik dari LFSR, f(x), adalah fungsi polinom berderajat 80. Fungsi dapat didefinisikan sebagai berikut:

$$f(x) = 1 + x^{18} + x^{29} + x^{42} + x^{57} + x^{67} + x^{80} \tag{2.2}$$

Untuk menghapus segala keambiguan yang mungkin terjadi, Grain juga menyediakan fungsi *update* pada pada persamaan 2.3 LFSR sebagai berikut:

$$s_{i+80} = s_{i+62} + s_{i+51} + s_{i+38} + s_{i+23} + s_{i+13} + s_i \tag{2.3}$$

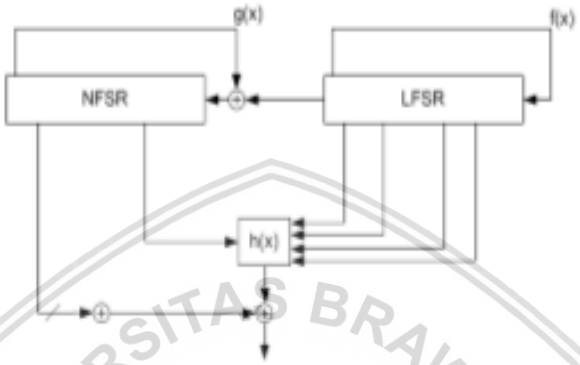
Fungsi umpan balik dari NFSR, g(x), didefinisikan sebagai berikut.

$$g(x) = 1 + x^{18} + x^{20} + x^{28} + x^{35} + x^{43} + x^{47} + x^{52} + x^{59} + x^{66} + x^{71} + x^{80} + x^{17}x^{20} + x^{43}x^{47} + x^{65}x^{71} + x^{20}x^{28}x^{35} + x^{47}x^{52}x^{59} + x^{17}x^{35}x^{52}x^{71} + x^{20}x^{28}x^{43}x^{47} + x^{17}x^{20}x^{59}x^{65} + x^{17}x^{20}x^{28}x^{35}x^{43} + x^{47}x^{52}x^{59}x^{65}x^{71} + x^{28}x^{35}x^{43}x^{47}x^{52}x^{59} \tag{2.4}$$

Fungsi update dari NFSR dapat didefinisikan sebagai berikut.

$$b_{i+80} = s_i + b_{i+62} + b_{i+60} + b_{i+52} + b_{i+45} + b_{i+37} + b_{i+33} + b_{i+28} + b_{i+21} + b_{i+14} + b_{i+9} + b_i + b_{i+63}b_{i+60} + b_{i+37}b_{i+33} + b_{i+15}b_{i+9} + b_{i+60}b_{i+52}b_{i+45} + b_{i+33}b_{i+28}b_{i+21} + b_{i+63}b_{i+45}b_{i+28}b_{i+9} + b_{i+60}b_{i+52}b_{i+37}b_{i+33} + b_{i+63}b_{i+60}b_{i+21}b_{i+15} + b_{i+63}b_{i+60}b_{i+52}b_{i+45}b_{i+37} + b_{i+33}b_{i+28}b_{i+21}b_{i+15}b_{i+9} + b_{i+52}b_{i+45}b_{i+37}b_{i+33}b_{i+28}b_{i+21} \quad (2.5)$$

Cara kerja Grain V1 secara umum digambarkan sebagai berikut:



Gambar 2.2 Diagram Umum Grain *Chiper*
Sumber : (Hell, Johansson, & Meier, 2006).

Isi dari kedua shift register merepresentasikan kondisi / state dari chiper. Dari kondisi ini, 5 variabel diambil sebagai masukan ke fungsi boolean h(x). Fungsi filter ini dipilih untuk seimbang, bebas korelasi dengan urutan pertama dan memiliki derajat aljabar 3. Ketidaklinearitasan adalah paling memungkinkan untuk fungsi ini. Masukan diambilbaik dari LFSR dan dari NFSR. Fungsi tersebut dapat didefinisikan sebagai berikut:

$$h(x) = x_1 + x_4 + x_0x_3 + x_2x_3 + x_3x_4 + x_0x_1x_2 + x_0x_2x_3 + x_0x_2x_4 + x_1x_2x_4 + x_2x_3x_4 \quad (2.6)$$

$$z_i = \sum_{k\in A}^{n} b_{i+k} + h(s_{i+3}, s_{i+25,}s_{i+46}, s_{i+64}, b_{i+63}) \quad (2.7)$$

Dimana *A* = {1, 2, 4, 10, 31, 43, 56}

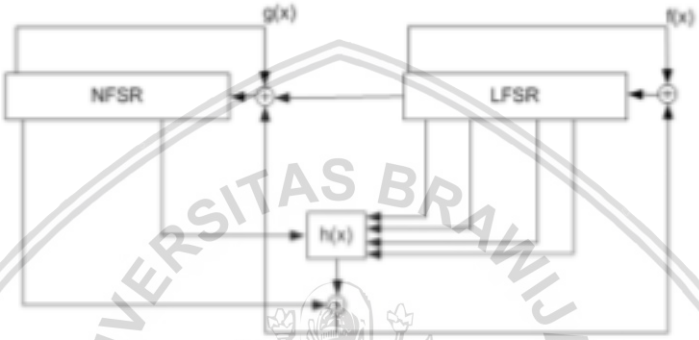
Pengubah x_0, x_1, x_2, x_3, dan x_4 sesuai dengan s_{i+3}, s_{i+25}, s_{i+46}, s_{i+64}, dan b_{i+63}. Fungsi filter ini kemudian di-*xor* dengan bit b_i dari NFSR untuk menghasilkan sebuah *keystream.*

2.2.3.2 Inisialisasi kunci

Sebelum tiap *keystream* dibangkitkan, chiper harus diinisialisasi dengan kunci dan IV (*Initial Value*). Misal bit bit dari kunci k, dilambangkan dengan k dengan 0 <= i <= 79, dan bit-bit dari IV dilambangkan dengan IV dengan 0 <= i <= 63.

Inisialisasi kunci dilakukan dengan cara yaitu :

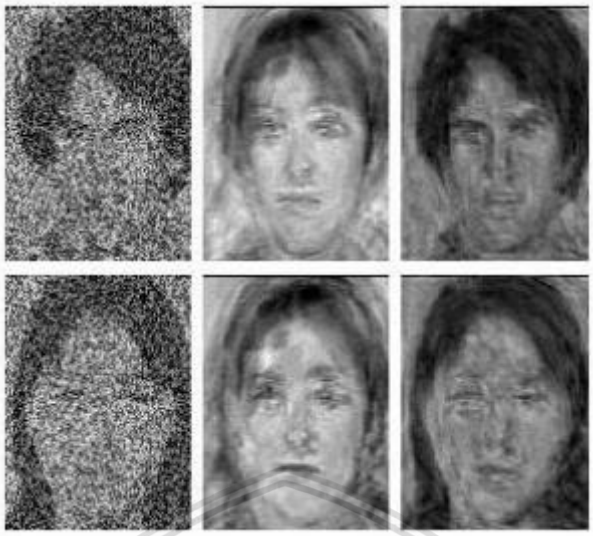
1. NFSR diisi dengan bit-bit kunci, b = k, 0 <=i <= 79
2. 64 bit pertama dari LFSR diisi dengan IV, s = IV 0 <= i <= 63
3. Sisa bit dari LFSR tidak dapat diinisialisasi dengan seluruhnya kondisi 0 (*zero state*).
4. Chiper di-*clock* 160 kali tanpa membuat satupun running key. Keluaran dari fungsi filter dikembalikan dan di-xor dengan masukan, baik ke LFSR dan ke NFSR, seperti pada Gambar 2.3 dibawah ini



Gambar 2.3 Inisialisasi Grain *Cipher*

2.2.4 *Image Proccesing*

Kata image yang berarti Gambar, memiliki banyak kegunaan dalam kehidupan sehari-hari. gambar memberikan suatu informasi, interpretasi, ilustrasi, evaluasi, komunikasi dan hiburan bagi kita. Image Processing adalah suatu bentuk pengolahan atau pemrosesan sinyal dengan input berupa gambar (image) dan ditransformasikan menjadi gambar lain sebagai keluarannya dengan teknik tertentu. Image processing dilakukan untuk memperbaiki kesalahan data sinyal gambar yang terjadi akibat transmisi dan selama akuisisi sinyal, serta untuk meningkatkan kualitas penampakan gambar agar lebih mudah diinterpretasi oleh sistem penglihatan manusia baik dengan melakukan manipulasi dan juga penganalisisan terhadap gambar. (Ade, 2009)



Gambar 2.4 Proses Perubahan Gambar

Dalam hal ini metode yang digunakan untuk melakukan proses image processing tersebut yaitu dengan menggunakan metode Algoritme Grain V1. Untuk melakukan perubahan Gambar tersebut cara yang digunakan yaitu dengan menggunakan bit per pixel sebagai dasar atau acuan dalam proses mekanisme enkripsi dari sebuah Gambar yang disediakan.

Bit per piksel mengambil konsep dasar dari bit biner dimana angka dapat diwakili oleh satu bit yaitu 0 dan 1, untuk dua kombinasi bit seperti 00 01 10 11. Rumus untuk perhitungan jumlah kombinasi yang dapat dibuat dari bit:

$$(2)^{bpp}$$

Dimana bpp menunjukan bit per piksel

Jumlah warna yang berbeda bergantung pada jumlah bit per piksel

1 bpp 2 colors
2 bpp 4 colors
3 bpp 8 colors
4 bpp 16 colors
5 bpp 32 colors
6 bpp 64 colors
7 bpp 128 colors
8 bpp 256 colors
10 bpp 1024 colors
16 bpp 65536 colors

Gambar warna biasanya dari format 24 bpp atau 16 bpp.

2.2.5 *Web Based Application*

Web based Application yaitu aplikasi yang dapat berjalan menggunakan basis teknologi web (internet) atau browser. Keunggulan-keunggulan yang dimiliki diantaranya yaitu

1. Kita dapat menjalankan aplikasi berbasis web dimanapun kapanpun tanpa harus melakukan penginstalan.
2. Terkait dengan isu lisensi (hak cipta), kita tidak memerlukan lisensi ketika menggunakan web-based application, sebab lisensi telah menjadi tanggung jawab dari web penyedia aplikasi.
3. Dapat dijalankan di sistem operasi manapun. Tidak perduli apakah kita menggunakan linux, windows, aplikasi berbasis web dapat dijalankan asalkan kita memiliki browser dan akses internet.
4. Dapat diakses lewat banyak media seperti: computer, handheld dan handphone yang sudah sesuai dengan standard WAP.
5. Tidak perlu spesifikasi computer yang tingggi untuk menggunakan aplikasi berbasis web ini, sebab di beberapa kasus, sebagian besar proses dilakukan di web server penyedia aplikasi berbasis web ini.

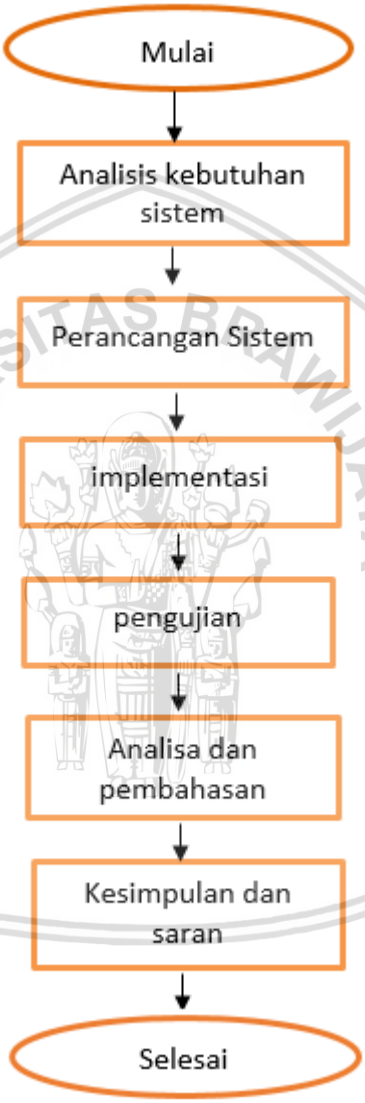
Kekurangan

1. Dibutuhkan koneksi intranet dan internet yang handal dan stabil, hal ini bertujuan agar pada saat aplikasi dijalankan akan berjalan dengan baik dan lancer.

Dibutuhkan sistem keamanan yang baik dikarenakan aplikasi dijalankan secara terpusat, sehingga apabila server di pusat *down* maka sistem aplikasi tidak bias berjalan.

BAB 3 METODOLOGI

Pada bab ini akan dijelaskan tentang metode penelitian yang akan digunakan dalam tahapan-tahapan penelitian dan penyusunan skripsi. Metode penelitian yang akan dilakukan pada penelitian ini dapat dilihat dari diagram alir pada gambar 3.1



Gambar 3. 1 Diagram Alir Metodologi Penelitian

Pada Gambar 3.1 merupakan diagram alir metode yang dimanfaatkan oleh penulis dalam menyusun penelitian.

3.1 Analisis Kebutuhan Sistem

Sub bab ini akan mengidentifikasi kebutuhan dalam pembangunan sistem ini. Kebutuhan yang digunakan adalah sebagai berikut:

a. Spesifikasi perangkat keras yang digunakan
   - Laptop ASUS X454W, AMD E1 CPU @ 1.35GHz
   - RAM 4 GB

b. Spesifikasi perangkat lunak yang digunakan
   - Sistem Operasi Linux Ubuntu 14.04
   - Apache Web Server
   - Redis Database
   - Sublime Text 3 Editor
   - Browser Google Chrome, Mozilla Firefox, dan Microsoft Edge

c. Kebutuhan data

Data diambil dari beberapa sample Gambar dengan kualitas rendah hingga kualitas tinggi.

3.2 Perancangan Sistem

Perancangan sistem dilakukan setelah tahap analisis kebutuhan selesai dilakukan. Terdapat dua langkah dalam perancangan sistem yaitu perancangan diagram alir serta perancangan desain tampilan dari aplikasi.

Perancangan diagram alir sistem keamanan aplikasi dilakukan untuk mengGambarkan proses perlindungan terhadap file dalam bentuk Gambar melalui proses enkripsi menggunakan algoritme kriptografi Grain.

Sedangkan perancangan desain tampilan aplikasi dilakukan untuk memudahkan dalam melakukan pembuatan *user interface* pada web, sehingga pada saat melakukan pengkodean, penulis sudah tahu bahwa aplikasi yang dibuat harus memiliki tampilan seperti apa.

3.3 Implementasi

Tahap ini akan menjelaskan bagaimana mengimplentasikan sistem berdasarkan perancangan sistem yang telah dibuat pada tahap sebelumnya. Sistem akan dibangun menggunakan bahasa pemrograman PHP dengan bantuan sesuai yang telah dipaparkan pada sub bab analisis kebutuhan. Pada bab ini juga mengimplementasikan algoritme Grain chiper versi 1, bagaimana melakukan proses enkripsi dengan menggunakan algoritme Grain chiper untuk proses image processing.

3.4 Pengujian

Pada tahap ini, dilakukan pengujian pada sistem yaitu pengujian pada perangkat lunak dapat berjalan dengan baik atau tidaknya sesuai dengan spesifikasi, tujuan, dan kebutuhan yang telah dianalisis sebelumnya untuk mengetahui kinerja sistem dari segi fungsionalitas dan performa sistem.

Selanjutnya pengujian test vector untuk mengetahui apakah sudah sesuai dengan acuan yang terdapat pada *paper* yang menjadi landasan dalam penelitian ini.

3.5 Analisis dan Pembahasan

Selanjutnya pada tahap ini dilakukan analisis dan pembahasan. hasil yang telah didapatkan berdasarkan implementasi menggunakan algoritme Grain cipher versi 1 dan pengujian yang telah dilakukan dengan indikator keberhasilan atau tidaknya suatu percobaan yaitu hasil berupa enkripsi dan dekripsi dari suatu objek Gambar akan dianalisa dan dilakukan pembahasan berdasarkan hasil pengujian yang telah dilakukan sebelumnya.

3.6 Kesimpulan dan Saran

Tahap akhir yaitu pengambilan kesimpulan dan saran merupakan hasil akhir dari setiap tahap yang telah dilakukan pada penelitian ini berdasarkan rumusan masalah yang telah disebutkan terlebih dahulu pada awal penelitian.

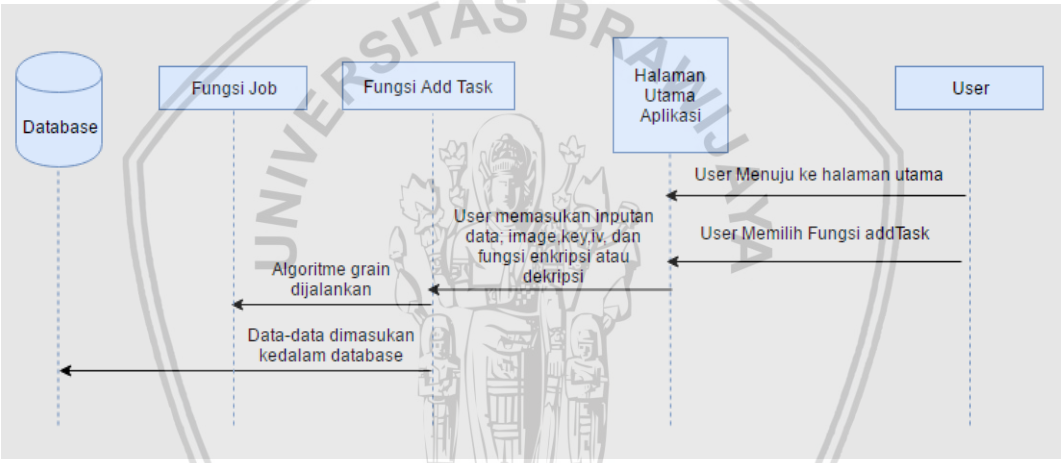
BAB 4 ANALISIS DAN PERANCANGAN

4.1 Analisis Kebutuhan

Pada analisis kebutuhan akan dibahas mengenai hal-hal apa saja yang dibutuhkan sistem enkripsi Gambar pada aplikasi berbasis web menggunakan algoritme Grain V1 sebagai proses enkripsi data. Analisis kebutuhan pada penelitian terdiri dari analisis permasalahan, analisis data dan analisis keamanan data yang nantinya akan dijelaskan pada sub bab selanjutnya.

4.1.1 Analisis Permasalahan

Pada analisis permasalahan akan dijelaskan mengenai proses kerja atau alur dari sistem enkripsi Gambar pada aplikasi berbasis web menggunakan algoritme Grain sebagai proses enkripsi data melalui Gambar 4.1 dibawah ini.



Gambar 4.1 Event Diagram Enkripsi Gambar Pada Aplikasi Berbasis Web

Penjelasan Event Diagram Enkripsi Gambar Pada Aplikasi Berbasis Web pada Gambar 4.1 diatas adalah sebagai berikut.

1. User membuka browser menuju ke halaman utama web
2. Kemudian user memilih fitur task yang terdapat pada halaman utama web yang berisikan objek images, key, initial value(iv), dan task
3. User menginputkan Gambar dengan menelusuri file Gambar pada folder tertentu diikuti dengan menginputkan key, initial value(iv) dan task. Pada fitur task terdapat dua opsi yaitu encrypt dan decrypt dimana user memilih opsi encrypt untuk melakukan proses enkrispi Gambar setelah hasil enkripsi kemudian user memilih opsi decrypt

4. Kemudian proses enkripsi berjalan dan setelah itu akan terlihat hasilnya dan selanjutnya melakukan proses dekripsi dengan cara yang sama seperti pada poin nomor 3.
5. Data-data tersebut disimpan didalam database.

Selanjutnya akan dijelaskan alur kerja dari algoritme Grain melalui Gambar 4.2 dibawah ini.

Gambar 4.2 Proses Kerja Algoritme Grain

Pertama, data yang akan diproses atau dilakukan enkripsi masih dalam bentuk *plaintext* kemudian sebelum dilakukannya proses enkripsi dilakukan proses inisialisasi key yang memiliki dua masukan yaitu key dan Initial Value (IV), kemudian dua data tersebut dirubah menjadi bilangan biner dan dimasukan kedalam blok fungsi NFSR untuk key dan LFSR untuk IV yang dimana masing-masing blok fungsi memiliki daya tampung maksimal 80 bit. Pada proses ini dilakukan permutasi Grain dan di clock sebanyak 160 kali

Selanjutnya data yang akan dienkripsi sebelumnya dirubah ke bilangan biner dan dihitung panjangnya, kemudian dilakukan permutasi Grain kembali sebanyak panjang dari data yang akan dienkripsi dengan nilai blok fungsi NFSR dan LFSR yang sudah diacak sebanyak 160 kali pada tahap sebelumnya. Pada tahap ini setiap clock yang dilakukan akan diambil nilainya dan dimasukan kedalam suatu variabel sehingga nilai dari variabel tersebut memiliki panjang yang sama dengan panjang data yang akan dienkripsi, nilai variabel tersebut lalu dilakukan perhitungan XOR dengan nilai binary dari data yang akan dienkripsi tadi sehingga terbentuklah ciphertext.

4.1.2 Analisis Data

Setelah tahap analisis permasalahan selanjutnya melakukan analisis terhadap data. Sesuai dengan kebutuhan dan topik penelitian, pada sistem ini data yang akan dilakukan proses enkripsi yaitu data berupa inputan Gambar yang dimasukan oleh user untuk melakukan proses enkripsi dan juga dekripsi di dalam aplikasi berbasis web.

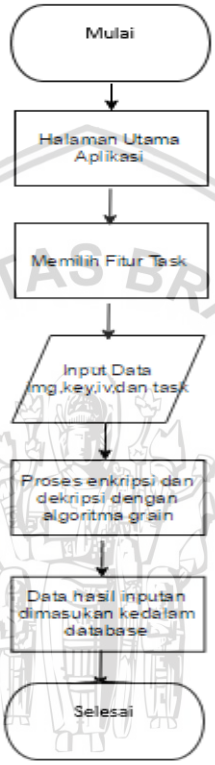
4.1.3 Analisis Keamanan Data

Tahap akhir pada analisis kebutuhan ini yaitu analisis keamanan data. Pada enkripsi Gambar berbasis website ini, data yang akan dilindungi kerahasiaannya yaitu data file dengan format Gambar. Penulis memutuskan untuk melakukan

proses enkripsi pada data file berupa Gambar-Gambar dari masukan user. Hasil enkripsi yang telah dilakukan akan membuat data tersebut menjadi data yang tidak dapat terbaca oleh siapapun kecuali pengguna yang sah karena data tersebut telah memenuhi sifat rahasia.

4.2 Perancangan Sistem

4.2.1 *Flowchart* Sistem



Gambar 4.3 Flowchart Enkripsi Gambar Pada Aplikasi Berbasis Web

Flowchart merupakan Gambaran Enkripsi Gambar Pada Aplikasi Berbasis Web dengan menerapkan algoritme *Grain* sebagai proses enkripsi data. Dibawah ini adalah proses kerja dari *flowchart* pada Gambar 4.3

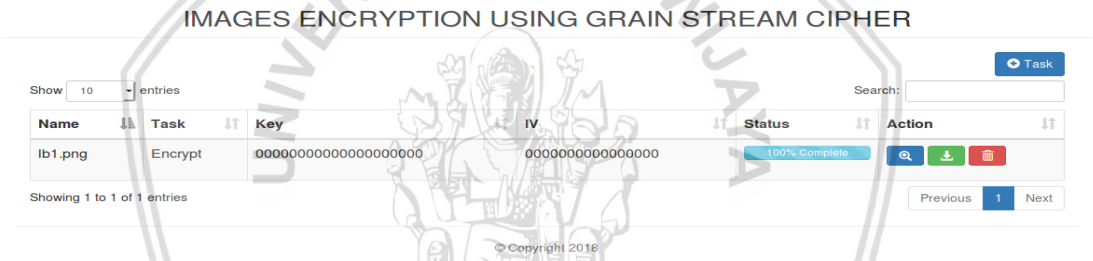
1. Mulai
2. Masuk ke halaman utama aplikasi
3. Selanjutnya fitur task yang akan mengerjakan beberapa proses kerja yang terdapat di dalam sistem.
4. Setelah memilih fitur task kemudian user menginputkan beberapa data untuk memulai proses kerja seperti upload data, memasukan key dan Initial Value (IV) serta memilih task yang akan dikerjakan apakah itu enkripsi atau dekripsi

5. Data akan diproses setelah user memilih task apa yang ingin dikerjakan bersamaan dijalankan algoritme Grain sebagai mekanisme untuk melakukan enkripsi data.
6. Data dari hasil inputan yang didapatkan bersamaan dengan hasil dari proses kerja algoritme Grain akan dimasukan kedalam database sistem yang akan ditampilkan hasilnya berupa data hasil enkripsi atau dekripsi
7. Selesai.

4.2.2 Desain Tampilan

Fungsi dari perancangan desain tampilan ini adalah untuk mempermudah dalam proses implementasi , dan juga agar sistem yang dibuat lebih efektif sesuai dengan harapan . berikut merupakan rancangan desain tampilan.

1. Desain tampilan halaman utama



Gambar 4.4 Desain Tampilan Halaman Utama

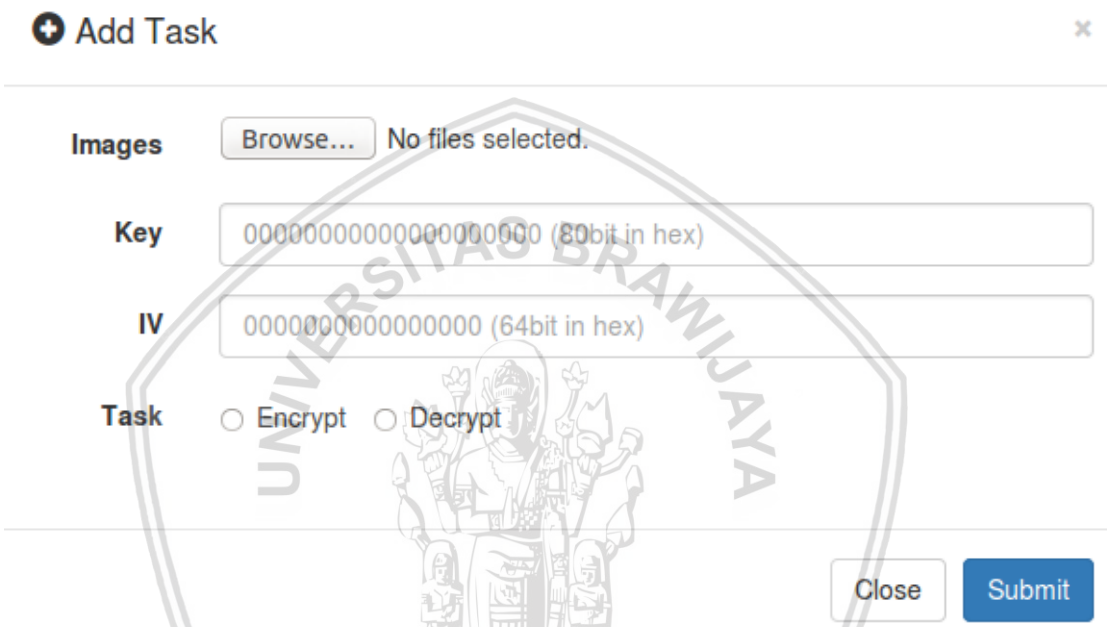
Berikut merupakan keterangan untuk desain tampilan halaman utama yang terdapat pada Gambar 4.4.

1. Fitur Pada Tabel yang berisikan

- Nama Gambar
- Task (Encrypt) : Melakukan Proses Enkrispi
  (Decrypt) : Melakukan Proses Dekripsi)
- Key : hasil inputan key yang telah diinputkan oleh user
- Initial Value : hasil inputan iv yang telah diinputkan oleh user
- Status : menyatakan sukses atau tidaknya hasil task yang dikerjakan
- Action : Terdiri dari tombol view, download, dan delete
  View : Untuk menunjukan hasil enkripsi
  Download : Untuk mengambil file yang diperlukan
  Delete : Untuk menghapus file

2. Fitur Show : menampilkan hasil dengan jumlah tertentu pada tabel
3. Tombol Task : digunakan sebagai inputan dari user
4. Fungsi Search : untuk mencari file yang diinginkan user dengan keyword tertentu
5. Previous dan Next : digunakan untuk memperlihatkan tabel sebelum dan sesudahnya.

2. Desain Tampilan Fitur Task



Gambar 4.5 Desain Tampilan Fitur Task

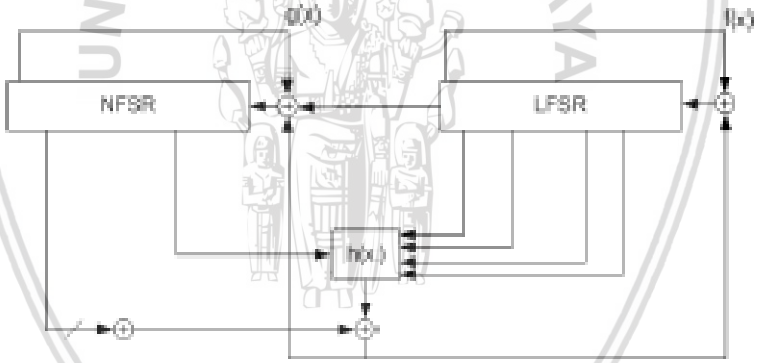
Berikut merupakan keterangan untuk desain tampilan halaman utama yang terdapat pada Gambar 4.5.

1. Inputan images untuk melakukan proses upload Gambar dengan menelusuri melalui direktori atau folder tertentu
2. Inputan key untuk menginputkan key dengan panjang key 80 bit
3. Inputan IV (Initial Value) untuk menginputkan IV dengan panjang 64 bit
4. Task untuk memilih proses enkripsi atau dekripsi
5. Radio button encrypt dan decrypt untuk melakukan proses enkripsi atau dekripsi
6. Tombol Close untuk menutup task
7. Tombol submit untuk melakukan proses akhir dari hasil inputan yang telah dimasukan

4.3 Perancangan Algoritme

Pada sub bab ini akan dijelaskan tahap-tahap dari perancangan algoritme Grain cipher melalui Gambar 4.6. Proses ini bertujuan untuk mendapatkan keystream yang nantinya akan dihasilkan cipher sebagai proses dari enkripsi. Sebelum tiap *keystream* dibangkitkan, cipher harus diinisialisasi dengan kunci (*key*) dan *IV* (Initial Value).

1. Untuk tiap bit dari key akan diinisialisasikan dengan 0 <= i <= 79 yang artinya berisikan dari indeks 0 sampai indeks ke-79, nilai default dari bit-bit tersebut bernilai 0.
2. Untuk tiap bit dari *IV* diinisialisasikan dengan 0 <=i <= 63 yang artinya berisikan dari indeks 0 sampai indeks ke-63, nilai default dari bit-bit tersebut bernilai 0.
3. Kemudian untuk tiap bit key dimasukan ke dalam NFSR dan untuk tiap bit *IV* dimasukan ke LFSR.
4. Sisa bit dari LFSR yaitu dari indeks ke-64 sampai indeks ke-79 diisikan dengan 1 atau diinisialisasikan dengan 64 <= i <= 79, LFSR tidak dapat diisikan dengan seluruhnya kondisi 0 (*zero state*).
5. Selanjutnya cipher di clock 160 kali tanpa membuat satupun keystream



Gambar 4.6 Inisialisasi Kunci

Tabel 4.1 *hx*

hx	**=** 00000000000000000011111111111000000100000011111111110000011100001000111100000110101111011010001011111110110110110011011000100100010100001100001000100001001100

Pada Tabel 4.1 merupakan perhitungan manual dari nilai *hx*. Ketika di *clock, hx* akan menghasilkan 1 bit nilai. Nilai tersebut masih berupa inisialisasi *key*.

Tabel 4.2 *zi*

zi	**=** 00000000000000000011111111111000000100000000000000000100110010 00010101110011100101101011101110010110000101110101001100110110 01010111001110110011101110100100001111

Pada Tabel 4.2 merupakan perhitungan manual dari nilai zi. Setiap nilai zi digunakan untuk proses inisialisasi *key* sehingga nilai *zi* tersebut belum berbentuk *keystream.*

Tabel 4.3 *fx*

fx	**=** 00111111111110000000111111000000000100111000000000110110100010 01100111101001101100100111010110100001100011011110011000101001 11010001001110111111011000100101101101 0

Pada Tabel 4.3 merupakan perhitungan manual dari nilai *fx.* nilai fx (untuk LFSR) akan menghasilkan 1 bit nilai pada saat proses clock dilakukan, tiap bit tersebut nantinya digunakan sebagai nilai untuk pengisian pada LFSR yaitu pada posisi indeks ke-79, nilai yang tertera didalam Tabel 4.3 diatas telah dilakukan xor dengan nilai pada zi.

Tabel 4.4 *gx*

gx	= 00000000000000000011111111111000000111111111111001110111100101 11001101111001010110110110101111011000001011001100110010001100 00101000101111000001111001011100100110

Pada Tabel 4.4 merupakan perhitungan manual dari nilai *gx.* nilai gx (untuk NFSR) akan menghasilkan 1 bit nilai pada saat proses clock dilakukan, tiap bit tersebut nantinya digunakan sebagai nilai untuk pengisian pada NFSR yaitu pada posisi indeks ke-79, nilai yang tertera didalam Tabel 4.4 diatas telah dilakukan xor dengan nilai pada zi.

Clock	Pergeseran LFSR
1	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0
2	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0
3	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1
4	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1
Dst...	..
160	0 1 0 0 1 1 1 0 1 0 1 1 0 1 0 0 0 0 1 1 0 0 0 1 1 0 1 1 1 1 0 0 1 1 0 0 0 1 0 1 0 0 1 1 0 1 0 0 0 1 0 0 1 1 1 0 1 1 1 1 1 0 1 1 0 0 0 1 0 0 1 0 1 1 0 1 1 0 1 0

Gambar 4.7 Pergeseran LFSR

Proses pengisian nilai 0 dari indeks 0 – 63, kemudian indeks ke 64 – 79 dilakukan pengisian nilai 1 kemudian akan bergeser 1 kali pada setiap clock nya hingga mencapai 160 clock. Pada Gambar 4.7 diatas diperoleh hasil pergeseran dari LFSR dimana pada indeks terakhir nilainya telah di subtitusikan dengan hasil dari bit fx dengan menghapus indeks pertama. LFSR tersebut telah melakukan pergesaran sebanyak 1 kali pada setiap clock nya.

Clock	Pergeseran NFSR
1	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
3	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
4	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Dst...	..
160	0 1 1 0 1 1 0 1 0 1 1 1 1 0 1 1 0 0 0 0 0 1 0 1 1 0 0 1 1 0 0 1 1 0 0 1 0 0 0 1 1 0 0 0 1 0 1 0 0 0 1 0 1 1 1 1 0 0 0 0 0 1 1 1 1 0 0 1 0 1 1 1 0 0 1 0 0 1 1 0

Gambar 4.8 Pergeseran NFSR

Pada Gambar 4.8 diatas diperoleh hasil pergeseran dari NFSR dimana pada indeks terakhir nilainya telah di subtitusikan dengan hasil dari bit *gx* dengan menghapus indeks pertama. NFSR tersebut telah melakukan pergesaran sebanyak 1 kali pada setiap clock nya.

Tabel 4.5 *hx*

hx	**=** 10110001000000011010001111001101010101111110010011000010010010011111110100001100

Pada tabel 4.5 telah dihasilkan nilai hx yang telah dilakukan proses clock sebanyak 161 atau sejumlah 80 kali, dari proses clock tersebut merupakan tahap awal dibuatnya keystream setelah proses clock sebanyak 160 kali.

Tabel 4.6 *zi*

zi	**=** 00000000000000000011111111111000000100000000000000001001100100001010111001110010110101110111001011000010111010100110011011010101110011101100111011101000100001111

Pada tabel 4.6 Kemudian diperoleh nilai zi yang hasilnya berupa keystream.

Tabel 4.7 *fx*

fx	**=** 11101000100111001010000100001001111010110111111100010100011011001101110100000000

Pada tabel 4.7 telah dihasilkan nilai fx yang berbeda dengan nilai pada saat tahap inisialisasi dimana kedua nilai tersebut diambil dari indeks ke 79 pada LFSR tanpa melakukan xor terhadap nilai zi.

Tabel 4.8 *gx*

gx	= 10001110100101010110011111000111111111010000010110001001110010010001011011111101

Pada tabel 4.8 ini telah dihasilkan nilai gx yang berbeda dengan nilai pada saat tahap inisialisasi dimana kedua nilai tersebut diambil dari indeks ke 79 pada NFSR tanpa melakukan xor terhadap nilai zi.

Clock	Pergeseran LFSR
1	1 0 0 1 1 1 0 1 0 1 1 0 1 0 0 0 0 1 1 0 0 0 1 1 0 1 1 1 1 0 0 1 1 0 0 0 1 0 1 0 0 1 1 0 1 0 0 0 1 0 0 1 1 1 0 1 1 1 1 1 0 1 1 0 0 0 1 0 0 1 0 1 0 1 1 0 1 1 0 1 0 1
2	0 0 1 1 1 0 1 0 1 1 0 1 0 0 0 0 1 1 0 0 0 1 1 0 1 1 1 1 0 0 1 1 0 0 0 1 0 1 0 0 1 1 0 1 0 0 0 1 0 0 1 1 1 0 1 1 1 1 1 0 1 1 0 0 0 1 0 0 1 0 1 1 0 1 1 0 1 0 1 1
3	0 1 1 1 0 1 0 1 1 0 1 0 0 0 0 1 1 0 0 0 1 1 0 1 1 1 1 0 0 1 1 0 0 0 1 0 1 0 0 1 1 0 1 0 0 0 1 0 0 1 1 1 0 1 1 1 1 1 0 1 1 0 0 0 1 0 0 1 0 0 1 1 0 1 1 0 1 0 1 1 1
Dst...	..
80	1 1 1 0 1 0 0 0 1 0 0 1 1 1 0 0 1 0 1 0 0 0 0 1 0 0 0 0 1 0 0 1 1 1 1 0 1 0 1 1 0 1 1 1 1 1 1 1 1 0 0 0 1 0 1 0 0 0 1 1 0 1 1 0 0 1 1 0 1 1 1 0 1 0 0 0 0 0 0 0 0

Gambar 4.9 Pergeseran LFSR

Pada Gambar 4.9 diperoleh hasil pergeseran dari LFSR dimana pada indeks terakhir nilainya telah di subtitusikan dengan hasil dari bit fx dengan menghapus indeks pertama. LFSR tersebut telah melakukan pergeseran sebanyak 1 kali pada setiap clock nya dengan perulangan 80 kali.

Clock	Pergeseran NFSR
1	1 1 0 1 1 0 1 0 1 1 1 1 0 1 1 0 0 0 0 0 1 0 1 1 0 0 1 1 0 0 1 1 0 0 1 0 0 0 1 1 0 0 0 1 0 1 0 0 0 1 0 1 1 1 1 0 0 0 0 0 1 1 1 1 0 0 1 0 1 1 1 0 0 1 0 0 1 1 0 1
2	1 0 1 1 0 1 0 1 1 1 1 0 1 1 0 0 0 0 0 1 0 1 1 0 0 1 1 0 0 1 1 0 0 1 0 0 0 1 1 0 0 0 1 0 1 0 0 0 1 0 1 1 1 1 0 0 0 0 0 1 1 1 1 0 0 1 0 1 1 1 0 0 1 0 0 1 1 0 1 0
3	0 1 1 0 1 0 1 1 1 1 0 1 1 0 0 0 0 0 1 0 1 1 0 0 1 1 0 0 1 1 0 0 1 0 0 0 1 1 0 0 0 1 0 1 0 0 0 1 0 1 1 1 1 0 0 0 0 0 1 1 1 1 0 0 1 0 1 1 1 0 0 1 0 0 1 1 0 1 0 0
Dst...	..
80	1 0 0 0 1 1 1 0 1 0 0 1 0 1 0 1 0 1 1 0 0 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 0 1 0 0 0 0 0 1 0 1 1 0 0 0 1 0 0 1 1 1 0 0 1 0 0 1 0 0 0 1 0 1 1 0 1 1 1 1 1 1 0 1

Gambar 4.10 Pergeseran NFSR

Pada Gambar 4.10 diperoleh hasil pergeseran dari NFSR dimana pada indeks terakhir nilainya telah di subtitusikan dengan hasil dari bit gx dengan menghapus indeks pertama. NFSR tersebut telah melakukan pergesaran sebanyak 1 kali pada setiap clock nya dengan perulangan 80 kali.

BAB 5 IMPLEMENTASI DAN PENGUJIAN

Implementasi adalah tahap penerapan terhadap sistem yang telah dirancang sebelumnya. Dalam penelitian ini, penulis menggunakan bahasa pemrograman php. Berdasarkan hasil analisis dan perancangan yang sudah dilakukan , akan dilakukan pembuatan enkripsi Gambar pada aplikasi berbasis web yang mengimplementasikan algoritme Grain dan menggunkan redis database sebagai database untuk menyimpan hasil dari pilihan user.

Pada bagian pengujian, akan dilakukan pengujian dengan beberapa jenis serangan terhadap sistem yang telah dibuat untuk memastikan tingkat confidentiality user dari aplikasi yang telah dibuat sebelumnya.

Tabel 5.1 Tabel Spesifikasi Perangkat Lunak dan Perangkat Keras

No.	Jenis Perangkat	Komponen
1.	Perangkat Keras	a. ASUS X454W, AMD E1 CPU @ 1.35GHz b. RAM 4 GB c. Harddisk dengan kapasitas 500 GB
2.	Perangkat Lunak	a. Operating System Ubuntu 14.04 LTS b. Sublime Text 3 Editor c. Redis Database d. Apache Web Server e. Web Browser

5.1 Implementasi

5.1.1 Algoritme Grain

5.1.1.1 Inisialisasi *Key*

Inisialisasi key digunakan untuk melakukan inisialisasi variabel dan fungsi yang akan digunakan selama proses enkripsi. Terdapat beberapa inisialisasi seperti LFSR dan NFSR diikuti dengan keysize yang diasumsikan dengan value sebanyak 80 bit key serta Initial Value (IV) sebanyak 64 bit IV dan juga terdapat NFTable yang digunakan untuk mengkalkulasikan feedback serta booltable yang digunakan untuk mengkalkulasikan output bits yang hasil akhirnya berupa keystream.

Algoritme 1: Source Code Inisialisasi Key

```
1  class Grain{
2
3      var $NFSR;
4
5      var $LFSR;
6
7      var $keysize = 80;
```

```
8       var $ivsize = 64;
9
10      var $NFTable = array(
11
12  0,1,0,0,1,0,1,1,1,0,1,1,0,1,0,0,1,0,1,1,0,1,0,0,0,1,0,0,0,1,0,1,
13
14
15  1,0,1,1,0,1,0,0,0,1,0,0,1,0,1,1,1,0,1,1,0,1,0,0,0,1,0,0,0,1,0,1,
16
17
18  1,0,1,1,0,1,0,0,0,1,0,0,1,0,1,1,0,1,0,0,1,0,1,1,1,0,1,1,1,0,1,0,
19
20  0,1,0,0,1,0,1,1,1,0,1,1,0,1,0,0,0,1,0,0,1,0,1,1,1,0,1,1,1,0,1,0,
21
22
23  1,0,1,1,0,1,0,0,0,1,0,0,1,0,1,1,0,1,0,0,1,0,1,1,1,0,1,1,1,0,1,0,
24
25  0,1,0,0,1,0,1,1,1,0,1,1,0,1,0,0,0,1,0,0,1,0,1,1,1,0,1,1,1,0,1,0,
26
27
28  0,1,0,0,1,0,1,1,1,0,1,1,0,1,0,0,1,0,1,1,0,1,0,0,0,1,0,0,0,1,0,1,
29
30  1,0,1,1,0,1,0,0,0,1,0,0,1,0,1,1,1,0,1,1,0,1,0,0,0,1,0,0,1,0,1,0,
31
32
33  1,0,1,1,0,1,0,0,0,1,0,0,1,0,1,1,0,1,0,0,1,0,1,1,1,0,1,1,1,0,1,0,
34
35
36  0,1,0,0,1,0,1,1,1,0,1,1,0,1,0,0,0,1,0,0,1,0,1,1,1,0,1,1,1,0,1,0,
37
38  0,1,0,0,1,0,1,1,1,0,1,1,0,1,0,0,1,0,1,1,0,1,0,0,0,1,0,0,0,1,0,1,
39
40
41  1,0,1,1,0,1,0,0,0,1,0,0,1,0,1,1,1,0,1,1,0,1,0,0,0,1,0,0,0,1,0,1,
42
43  0,1,0,0,1,0,1,1,1,0,1,1,0,1,0,0,1,0,1,1,0,1,0,0,0,1,0,0,0,1,0,1,
44
45
46  1,0,1,1,0,1,0,0,0,1,0,0,1,0,1,1,0,1,0,0,1,0,1,1,1,0,1,1,1,0,1,0,
47
48
49  0,1,0,0,1,0,1,1,1,0,1,1,0,1,0,0,1,0,1,1,0,1,0,0,0,1,0,0,0,1,0,1,
50
51  1,0,1,1,0,1,0,0,0,1,0,0,1,0,1,1,0,1,0,0,1,0,1,1,1,0,1,1,0,1,0,1,
52
53
54  0,1,0,0,1,0,1,1,1,0,1,1,0,1,0,0,1,0,1,1,0,1,0,0,0,1,0,0,0,1,0,1,
55
56  1,0,1,1,0,1,0,0,0,1,0,0,1,0,1,1,1,0,1,1,0,1,0,0,0,1,0,0,0,1,0,1,
57
58
59  1,0,1,1,0,1,0,0,0,0,0,1,1,1,1,0,0,1,0,0,1,0,1,1,1,1,1,0,1,1,1,1,
60
61
62  0,1,0,0,1,0,1,1,1,1,1,0,0,0,0,1,0,1,0,0,1,0,1,1,1,1,1,0,1,1,1,1,
63
64  1,0,1,1,0,1,0,0,0,1,0,0,1,0,1,1,0,1,0,0,1,0,1,1,1,0,1,1,1,0,1,0,
65
66
67  0,1,0,0,1,0,1,1,1,0,1,1,0,1,0,0,0,1,0,0,1,0,1,1,1,0,1,1,1,0,1,0,
68
69  0,1,0,0,1,0,1,1,1,1,1,0,0,0,0,1,1,0,1,1,0,1,0,0,0,0,0,1,0,0,0,0,
70
71
72  1,0,1,1,0,1,0,0,0,0,0,1,1,1,1,0,1,0,1,1,0,1,0,0,0,0,0,1,1,1,1,1,
```

```
73
74  0,1,0,0,1,0,0,0,1,0,1,1,0,1,1,1,1,0,1,1,0,1,1,1,0,1,0,0,0,1,1,0,
75
76
77  1,0,1,1,0,1,1,1,0,1,0,0,1,0,0,0,1,0,1,1,0,1,1,1,0,1,0,0,0,1,1,0,
78
79  1,0,1,1,0,1,1,1,0,0,0,1,1,1,0,1,0,1,0,0,1,0,0,0,1,1,1,0,1,1,0,0,
80
81
82  0,1,0,0,1,0,0,0,1,1,1,0,0,0,1,0,0,1,0,0,1,0,0,0,1,1,1,0,1,1,0,0,
83
84  1,0,1,1,0,1,1,1,0,1,0,0,1,0,0,0,0,1,0,0,1,0,0,0,1,0,1,1,1,0,0,1,
85
86
87  0,1,0,0,1,0,0,0,1,0,1,1,0,1,1,1,1,0,1,1,0,1,1,1,0,1,0,0,0,1,1,0,
88
89
90  1,0,1,1,0,1,1,1,0,0,0,1,1,1,0,1,0,1,0,0,1,0,0,0,1,1,1,0,1,1,0,0,
91
92  1,0,1,1,0,1,1,1,0,0,0,1,1,1,0,1,0,1,0,0,1,0,0,0,1,1,1,0,0,0,1,1
93
94      );
95
96      Var
97  $boolTable=array(0,0,1,1,0,0,1,0,0,1,1,0,1,1,0,1,1,1,0,0,1,0,1,1,0,1
    ,1,0,0,1,0,0);
```

5.1.1.2 Konversi String key dan Initial Value

Menkonversi key dan iv yang berbentuk string yaitu dari bentuk heksadesimal ke biner array (key in int). Dimana pada bagian ini terdapat fungsi str_split () yang digunakan untuk membagi atau mengubah string ke dalam array yaitu dengan mengubah key dan iv yang memiliki panjang potongan 2. Jika panjangnya kurang dari 1, fungsi str_split () akan mengembalikan FALSE tetapi Jika panjang lebih besar dari panjang string, seluruh string akan dikembalikan sebagai satu-satunya elemen dari array dalam hal ini panjang array yang diasumsikan 2. Selanjutnya fungsi array_map yang digunakan untuk mengirim setiap nilai string key dan iv dari sebuah array ke fungsi construct yang telah dibuat sebelumnya untuk mengembalikan nilai baru yang nantinya setiap nilai array ke fungsi tersebut akan dikalikan setiap nilai dengan sendirinya, dan dikembalikan dengan nilai-nilai baru. Kemudian yang terakhir dilakukan setup up load register pada key dan iv.

Algoritme 2: Source Code Konversi String Key dan Initial Value

```
1   function __construct($key, $iv){
2
3           // Convert string key  and iv of hex 0000000 to array(key in
4   int)
5           $inthex = function($value){
6
7               return hexdec($value);
8
9           };
10
11
12          $key = str_split($key, 2);
13          $key = array_map($inthex,$key);
14
```

```
15
16
17          $iv = str_split($iv, 2);
18          $iv = array_map($inthex,$iv);
19
20
21          $this->NFSR = array_fill(0, 80, 0);
22
23          $this->LFSR = array_fill(0, 80, 0);
24
25
26          //print_r($this->NFSR);
27
28
29          // key and iv setup load register
30
31          for($i=0; $i < $this->ivsize / 8; $i++){
32
33              for($j=0; $j < 8; $j++){
34                  $this->NFSR[$i * 8 + $j] = ($key[$i] >> $j) & 1;
35
36                  $this->LFSR[$i * 8 + $j] = ($iv[$i] >> $j) & 1;
37              }
38          }
39
40
41          for($i=$this->ivsize/8; $i < $this->keysize / 8; $i++){
42              for($j=0; $j < 8; $j++){
43
44                  $this->NFSR[$i * 8 + $j] = ($key[$i] >> $j) & 1;
45
46                  $this->LFSR[$i * 8 + $j] = 1;
47              }
48
49          }
50
```

5.1.1.3 Pembuatan Nilai Keystream

Setelah dilakukan konversi string key dan iv selanjutnya pada proses pembuatan nilai keystream dimulai dengan melakukan clock sebanyak 160 kali termasuk clock untuk LFSR dan NFSR untuk menghasilkan output bit yang mana output bit nantinya akan dihasilkan keystream. Kemudian terdapat Filter *function* yang mengambil beberapa nilai dari LFSR dan juga 1 nilai dari NFSR. Proses filter *function* mengambil beberapa nilai dari LFSR dan NFSR yang selanjutkan nilai tersebut akan dilakukan perhitungan dengan rumus h(x) dan menghasilkan suatu nilai keluaran dimana dalam kode program yang dimaksud yaitu kalkulasi feedback dan output bits yang telah di xor kan, kemudian dilakukan update register untuk menghasilkan output bit dimana pada fungsi akhir yaitu fungsi keystream bytes akan dihasilkan keluaran berupa keystream yang telah dibangkitkan.

Algoritme 3: Source Code Pembuatan Nilai Keystream

```
1   // Initial Clockint
2
3           for($i=0; $i < 160; $i++){
4               $outbit = $this->keystream();
5
```

```
            $this->LFSR[79] ^= $outbit;

            $this->NFSR[79] ^= $outbit;

        }

        //print_r($this->NFSR);

    }
//filter function

    private function N($i){

        return $this->NFSR[80 - $i];

    }

    private function L($i){

        return $this->LFSR[80 - $i];

    }

    public function keystream(){

        $X0 = $this->LFSR[3];

        $X1 = $this->LFSR[25];

        $X2 = $this->LFSR[46];

        $X3 = $this->LFSR[64];

        $X4 = $this->NFSR[63];


// Calculate feedback and output bits

        $outbit    =    $this->N(79)^$this->N(78)^$this->N(76)^$this->N(70)^$this->N(49)^$this->N(37)^$this->N(24)^$this->boolTable[($X4<<4) | ($X3<<3) | ($X2<<2) | ($X1<<1) | $X0];

        $NBit     =     $this->L(80)^$this->N(18)^$this->N(66)^$this->N(80)^$this->NFTable[($this->N(17)<<9)    |    ($this->N(20)<<8)    |    ($this->N(28)<<7)  |  ($this->N(35)<<6)  |  ($this->N(43)<<5)  |  ($this->N(47)<<4)   |   ($this->N(52)<<3)   |   ($this->N(59)<<2)   |   ($this->N(65)<<1) | $this->N(71)];

        $LBit     =     $this->L(18)^$this->L(29)^$this->L(42)^$this->L(57)^$this->L(67)^$this->L(80);


/ Update register

        for($i=1; $i < $this->keysize; $i++){

            $this->NFSR[$i-1] = $this->NFSR[$i];

            $this->LFSR[$i-1] = $this->LFSR[$i];

        }


        $this->NFSR[$this->keysize - 1] = $NBit;

        $this->LFSR[$this->keysize - 1] = $LBit;


        return $outbit;

    }


    public function keystream_bytes($msglen){
```

```
71          $keystream = array_fill(0, $msglen, 0);
72
73
74          for($i=0; $i < $msglen; $i++){
75
76              for($j=0; $j < 8; $j++){
77                  $outbit = $this->keystream();
78
79                  $outbit <<= $j;
80
81                  $keystream[$i] = $keystream[$i] | $outbit;
82              }
83
84          }
85
86
87          return $keystream;
88      }
```

5.1.1.4 Enkripsi

Proses enkripsi akan dikerjakan ketika sudah mendapatkan nilai *keystream* kemudian pada proses tersebut dilakukan perhitungan XOR terhadap nilai *keystream* dan nilai *binary* dari *plain text* sehingga nantinya akan dihasilkan *cipher text*.

Algoritme 4: Source Code Enkripsi

```
1   public function encrypt($msg){
2
3           $keystream = $this->keystream_bytes(sizeof($msg));
4           $cipher = array();
5
6
7           for($i=0; $i<sizeof($msg); $i++){
8
9               $cipher[] = $keystream[$i] ^ $msg[$i];
10          }
11
12
13
14          return $cipher;
```

5.1.1.5 Dekripsi

Dekripsi adalah proses dimana kita mengembalikan *cipher text* menjadi *plain text* dengan cara melakukan perhitungan XOR nilai *cipher text* terhadap nilai *keystream*.

Algoritme 5: Source Code Dekripsi

```
1   public function decrypt($msg){
2
3           $keystream = $this->keystream_bytes(sizeof($msg));
4           $plain = array();
5
6
7           for($i=0; $i<sizeof($msg); $i++){
8
9               $plain[] = $keystream[$i] ^ $msg[$i];
10
```

```
11          }
12
13
14          return $plain;
        }
```

5.1.2 Implementasi Pada Enkripsi Gambar

Mekanisme Enkripsi Gambar Pada Aplikasi Berbasis Web dengan menggunakan algoritme Grain cipher sebagai proses untuk merahasiakan data berupa file Gambar yang akan di upload pada fitur task. Dalam melakukan enkripsi Gambar ini digunakan konsep bit per pixel, bit diambil dari komponen Gambar yaitu RGB (red,green,blue) yang terdiri dari Merah, hijau dan biru dimana masing-masing menggunakan 8 bit yang memiliki nilai integer dari 0 hingga 255. Ini membuat 256 * 256 * 256 = 16777216 warna yang memungkinkan. Setelah set up untuk mendapatkan nilai bit dari RGB telah dilakukan pada fungsi perform kemudian proses enkripsi dijalankan dengan menyertakan file php berisi algoritme Grain cipher. Setelah itu dilakukan inisialisasi untuk menghasilkan warna baru dengan mengalokasikan sebuah warna dan menyimpannya ke dalam sebuah variabel. Warna baru inilah yang disebut sebagai image processing yaitu suatu bentuk pengolahan atau pemrosesan sinyal dengan input berupa Gambar (image) dan ditransformasikan menjadi Gambar lain sebagai keluarannya dengan teknik tertentu dalam hal ini teknik yang digunakan adalah teknik enkripsi.

Algoritme 6: Source Code Implementasi Pada Ekripsi Gambar

```
1   <?php
2
3   require 'Grain.php';
4
5
6   public function perform(){
7           $img = imagecreatefrompng($this->imgdir);
8
9           $x = imagesx($img);
10
11          $y = imagesy($img);
12
13          $total = ($x) * ($y);
14
15          $progress = 0;
16
17          if($this->task == 'Encrypt'){
18
19              for($i = 0; $i < $x; $i++){
20
21                  for($j = 0; $j < $y; $j++){
22
23                      $rgb = imagecolorat($img, $i, $j);
24
25                      $r = ($rgb >> 16) & 0xFF;
26
27                      $g = ($rgb >> 8) & 0xFF;
28
29                      $b = $rgb & 0xFF;
30
```

```
                    $rgb = $this->Grain->encrypt($rgb);

                    $new_color = imagecolorallocate($img, $rgb[0], $rgb[1], $rgb[2]);

                    imagesetpixel($img, $i, $j, $new_color);

                }


                $progress = ceil((($i+1) * ($y)) / $total) * 100);

                $sse = array('id' => $this->imgid, 'progress' => $progress);


                // Update status images on sqlite

                $qry = $this->db->prepare('UPDATE Job SET status = ? WHERE id = ?');

                $qry->execute(array((string)$progress,       $this->imgid));


                $fp = fopen('progress.json', 'w');

                fwrite($fp, json_encode($sse));

                fclose($fp);

                //echo "Proccesing " . $i;

            }

            $this->new_dir = 'assets/images/encrypted/' . time() . $this->imgname;

            imagepng($img, $this->new_dir);

            imagedestroy($img);

        }

        else{

            for($i = 0; $i < $x; $i++){

                for($j = 0; $j < $y; $j++){

                    $rgb = imagecolorat($img, $i, $j);

                    $r = ($rgb >> 16) & 0xFF;

                    $g = ($rgb >> 8) & 0xFF;

                    $b = $rgb & 0xFF;


8182

                    $rgb = array($r, $g, $b);

                    $rgb = $this->Grain->decrypt($rgb);

                    $new_color = imagecolorallocate($img, $rgb[0], $rgb[1], $rgb[2]);

                    imagesetpixel($img, $i, $j, $new_color);

                }

                $progress = ceil((($i+1) * ($y)) / $total) * 100);

                $sse = array('id' => $this->imgid, 'progress' => $progress);
```

```
                    // Update status images on sqlite
                    $qry = $this->db->prepare('UPDATE Job SET status = ? WHERE id = ?');
                    $qry->execute(array((string)$progress,       $this->imgid));


                    $fp = fopen('progress.json', 'w');
                    fwrite($fp, json_encode($sse));
                    fclose($fp);
                }
                $this->new_dir = 'assets/images/decrypted/' . time() . $this->imgname;
                imagepng($img, $this->new_dir);
                imagedestroy($img);
            }
        }
```

5.2 Parameter Pengujian

Pada penelitian ini menggunakan beberapa parameter pengujian untuk menguji algoritme Grain.

1. Pengujian Validitas *Test Vector*
2. Pengujian Fungsionalitas Sistem
3. Pengujian Kinerja Waktu
4. Pengujian Keamanan

5.3 Test Vector

5.3.1 Tujuan Pengujian

Tujuan pengujian *test vector* ialah memastikan bahwa algoritme Grain yang telah dibuat oleh penulis memiliki *output keystream* yang sama seperti dengan algoritme Grain yang telah dibuat oleh penciptanya.

5.3.2 Prosedur Pengujian

Prosedur pengujian *test vector* dilakukan dengan cara memberikan masukan yang telah ditentukan pada *test vector* dari algoritme Grain kemudian mencocokan *output* yang dihasilkan, pada algoritme Grain mencocokan nilai *keystream*. *Test vector* yang digunakan pada pengujian ini merujuk pada paper berjudul Grain - *A Stream Cipher for Constrained Environments.*

5.3.3 Hasil dan Analisis

5.3.3.1 Algoritme Grain

Test vector untuk algoritme Grain.

Key : 00000000000000000000
IV : 0000000000000000
Keystream : 7b978cf36846e5f4ee0b

Key : 0123456789abcdef1234
IV : 0123456789abcdef
Keystream : 42b567ccc65317680225

Sumber : (Hell, Johansson, & Meier, 2006)

Tabel 5.2 Validitas *Test Vector*

No.	Key (80 bit)	IV (64 bit)	Test Vector	Keystream	Hasil
1	00000000000000000000	0000000000000000	7b978cf36846e5f4ee0b	7b978cf36846e5f4ee0b	Valid
2	0123456789abcdef1234	0123456789abcdef	42b567ccc65317680225	42b567ccc65317680225	Valid

Dari kedua hasil percobaan diatas membuktikan bahwa hasil *keystream* yang digunakan oleh penulis dalam penelitian ini memiliki hasil *keystream* yang sama pada *test vector* yang telah disediakan pada algoritme Grain, sehingga bersifat *valid.*

5.4 Pengujian Fungsionalitas Sistem

5.4.1 Tujuan Pengujian

Dari kedua hasil percobaan diatas membuktikan bahwa hasil *keystream* yang digunakan oleh penulis dalam penelitian ini memiliki hasil *keystream* yang sama pada *test vector* yang telah disediakan pada algoritme Grain, sehingga bersifat *valid.*

5.4.2 Prosedur Pengujian

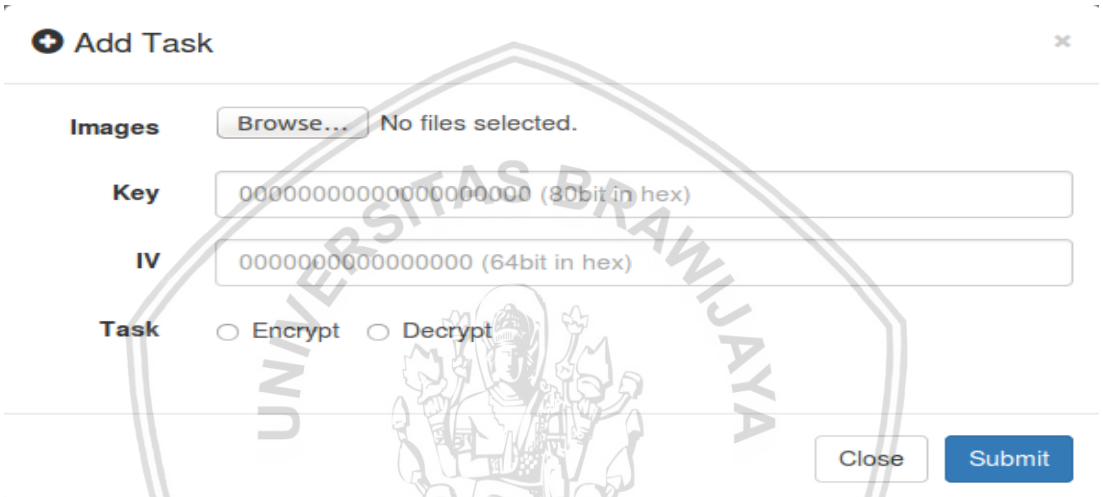
Prosedure pengujian fungsionalitas sistem dilakukan dengan cara melakukan test terhadap beberapa fungsi yang akan dijalankan yaitu dengan menggunakan pengujian unit testing dimana pengujian unit testing merupakan metode verifikasi dan validasi perangkat lunak dimana programmer menguji suatu unit program layak untuk tidaknya dipakai. Unit testing ini fokusnya pada verifikasi pada unit yang terkecil pada desain perangkat lunak, unit kecil ini dapat berupa fungsi atau prosedur.

5.4.3 Hasil Pengujian

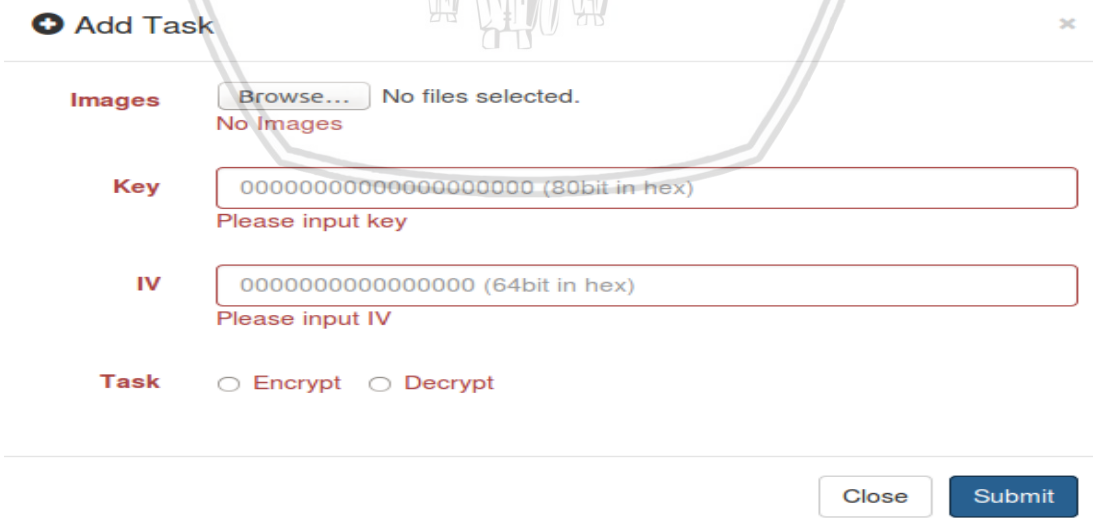
Berikut merupakan hasil pengujian fungsionalitas sistem Implementasi Algoritme Grain Cipher Untuk Enkripsi Gambar Pada Aplikasi Berbasis Web.

Tabel 5.3 Method AddTask (Validasi Input)

No.	Fungsi	Data Input	Expected Result	Result	Status
1	Semua Inputan Pada AddTask	Set validasi input = false, set error = true, tampilkan error = true	Jika validasi input sama dengan error maka menampilkan error.	Menampilkan pesan error jika validasi salah	Valid



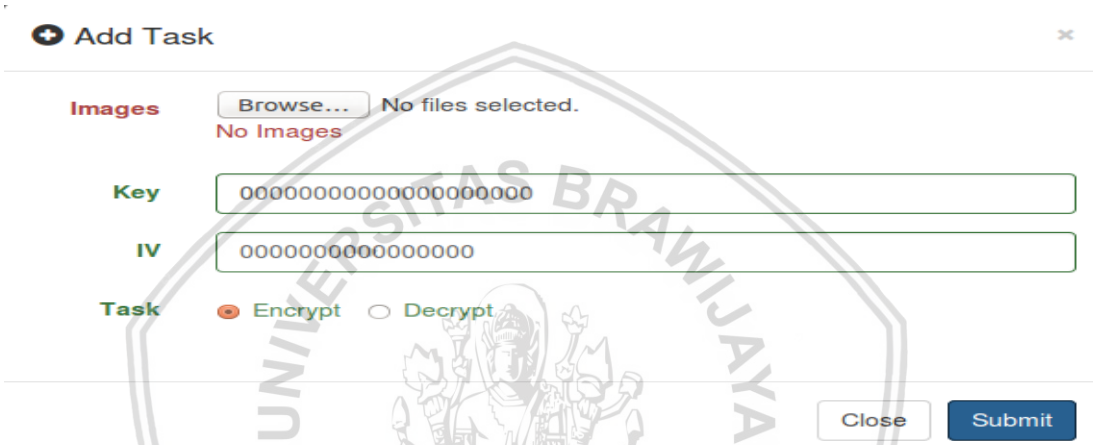
Gambar 5.1 Validasi Fungsi *AddTask*



Gambar 5.2 Hasil Validasi Fungsi *AddTask*

Tabel 5.4 Inputan *Images* (Validasi Input)

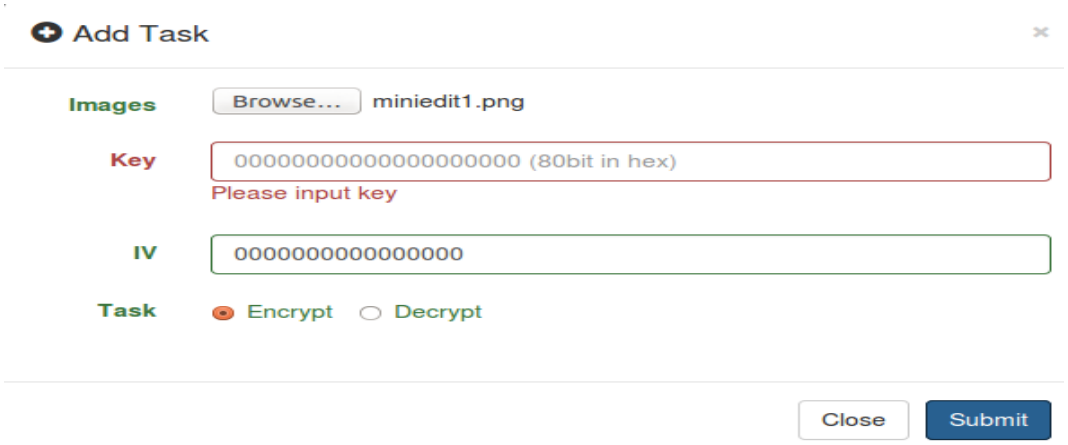
No.	Fungsi	Data Input	Expected Result	Result	Status
2	Inputan images	Set validasi input = false, set error = true, tampilkan error = true	Jika validasi input sama dengan error maka menampilkan error.	Menampilkan pesan error jika validasi inputan images tidak diinputkan	Valid



Gambar 5.3 Validasi Fungsi *images*

Tabel 5.5 Inputan *Key* (Validasi Input)

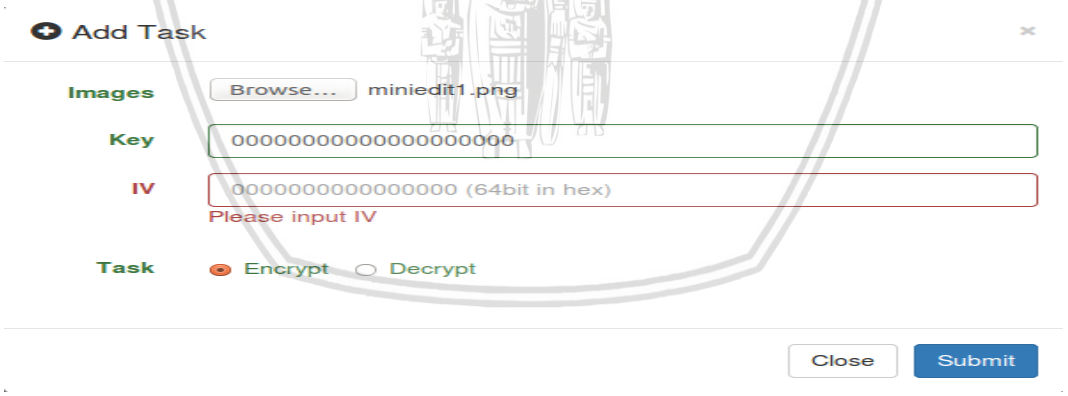
No.	Fungsi	Data Input	Expected Result	Result	Status
3.	Inputan Key	Set validasi input = false, set error = true, tampilkan error = true	Jika validasi input sama dengan error maka menampilkan error.	Menampilkan pesan error jika validasi input key tidak diinputkan	Valid



Gambar 5.4 Validasi Fungsi *key*

Tabel 5.6 Inputan IV (Validasi Input)

No.	Fungsi	Data Input	Expected Result	Result	Status
4.	Inputan IV	Set validasi input = false, set error = true, tampilkan error = true	Jika validasi input sama dengan error maka menampilkan error.	Menampilkan pesan error jika validasi input IV tidak diinputkan	Valid



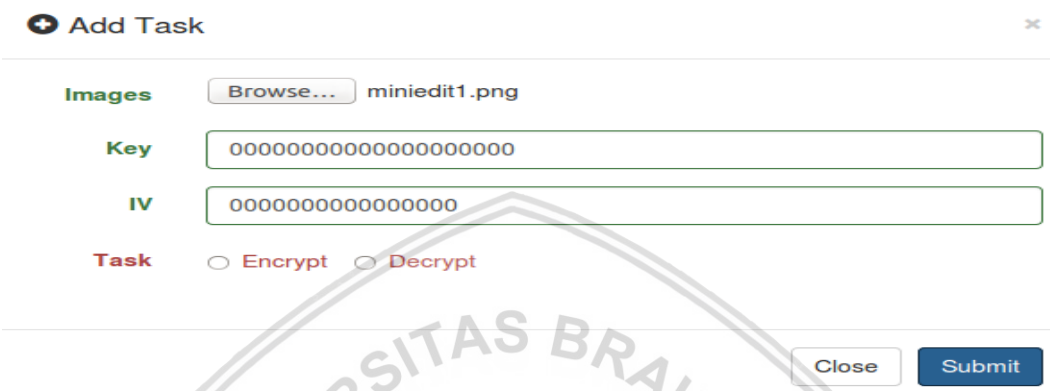
Gambar 5.5 Validasi Fungsi *InitialValue*

Tabel 5.7 Inputan Task (Validasi Input)

No.	Fungsi	Data Input	Expected Result	Result	Status
5.	Inputan Task	Set validasi input = false, set error = true, tampilkan	Jika validasi input sama dengan error maka	Menampilkan pesan error jika validasi input task	Valid

Tabel Lanjutan 5.7 Inputan Task (Validasi Input)

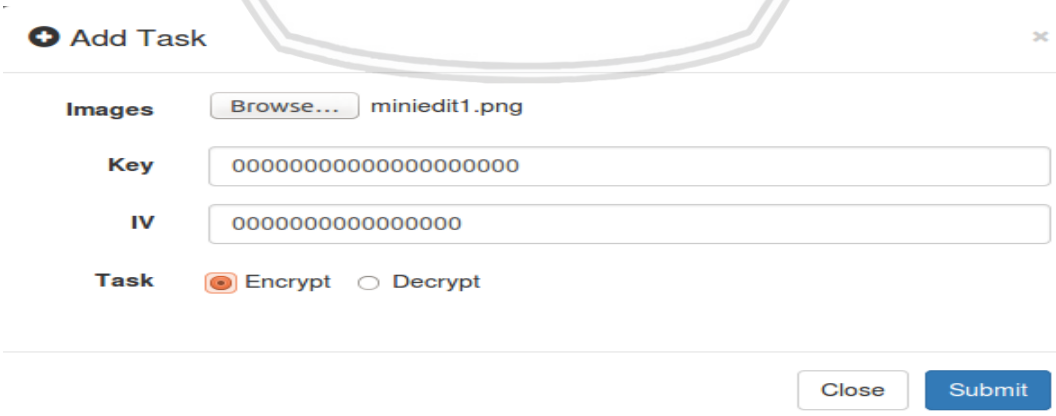
No.	Fungsi	Data Input	Expected Result	Result	Status
		error = true	menampilkan error.	tidak diinputkan	



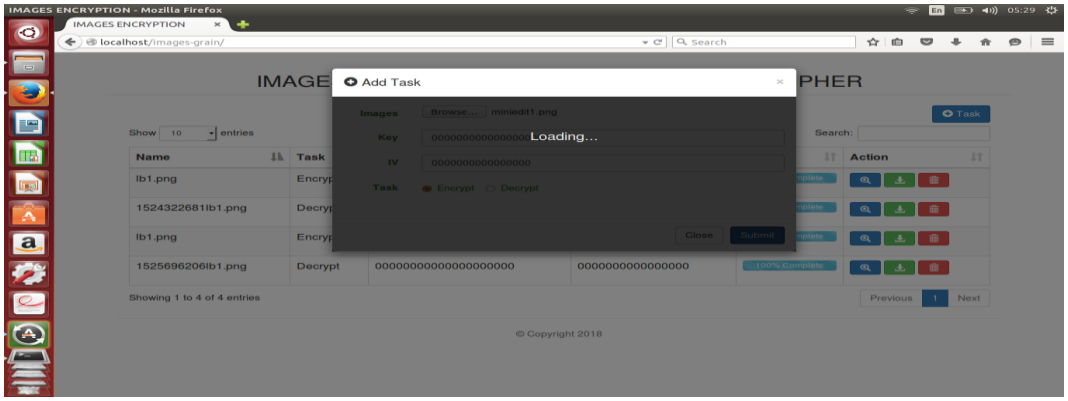
Gambar 5.6 Validasi Fungsi *Task*

Tabel 5.8 AddTask (Validasi Input)

No.	Fungsi	Data Input	Expected Result	Result	Status
6.	Semua Inputan Pada AddTask	Set validasi input = true, set error = false, tampilkan error = false	Jika validasi input benar maka proses dieksekusi.	Menampilkan proses loading jika validasi benar	Valid



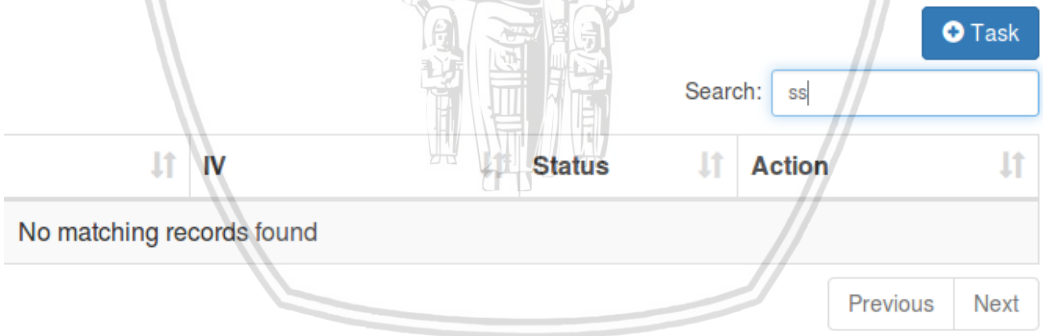
Gambar 5.7 Validasi Fungsi *Content AddTask*



Gambar 5.8 Validasi Fungsi Proses

Tabel 5. 9 Method Search

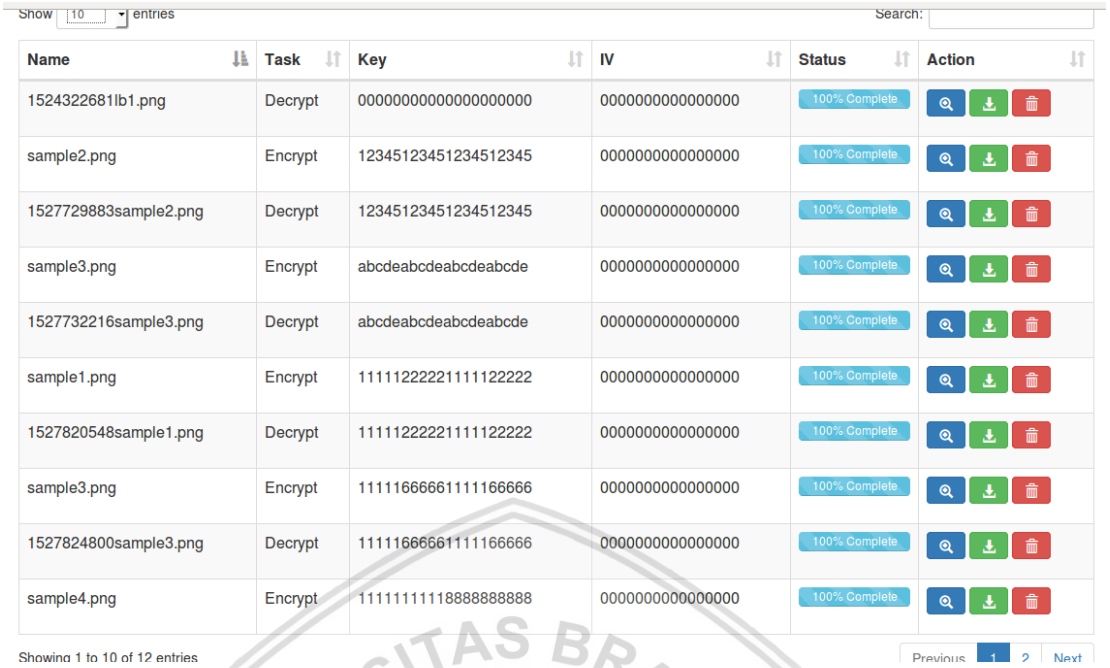
No.	Fungsi	Data Input	Expected Result	Result	Status
1	Inputan Search	Set validasi input = false, set error = true, tampilkan error = true	Jika validasi input sama dengan error maka menampilkan error.	Menampilkan pesan data tidak ditemukan	Valid



Gambar 5.9 Validasi Fungsi *Search*

Tabel 5. 10 Method Show Entries

No.	Fungsi	Data Input	Expected Result	Result	Status
1	Show Entries	Set validasi input = true, set error = false, tampilkan error = false	Jika validasi input benar maka proses dieksekusi.	Menampilkan data berdasarkan jumlah yang diinputkan	Valid


Show 10 entries Search:

Name	Task	Key	IV	Status	Action
1524322681lb1.png	Decrypt	00000000000000000000	00000000000000000	100% Complete	
sample2.png	Encrypt	12345123451234512345	00000000000000000	100% Complete	
1527729883sample2.png	Decrypt	12345123451234512345	00000000000000000	100% Complete	
sample3.png	Encrypt	abcdeabcdeabcdeabcde	00000000000000000	100% Complete	
1527732216sample3.png	Decrypt	abcdeabcdeabcdeabcde	00000000000000000	100% Complete	
sample1.png	Encrypt	11111222221111122222	00000000000000000	100% Complete	
1527820548sample1.png	Decrypt	11111222221111122222	00000000000000000	100% Complete	
sample3.png	Encrypt	11111666661111166666	00000000000000000	100% Complete	
1527824800sample3.png	Decrypt	11111666661111166666	00000000000000000	100% Complete	
sample4.png	Encrypt	11111111118888888888	00000000000000000	100% Complete	

Showing 1 to 10 of 12 entries Previous 1 2 Next

Gambar 5.10 Validasi Fungsi *Show Entries*

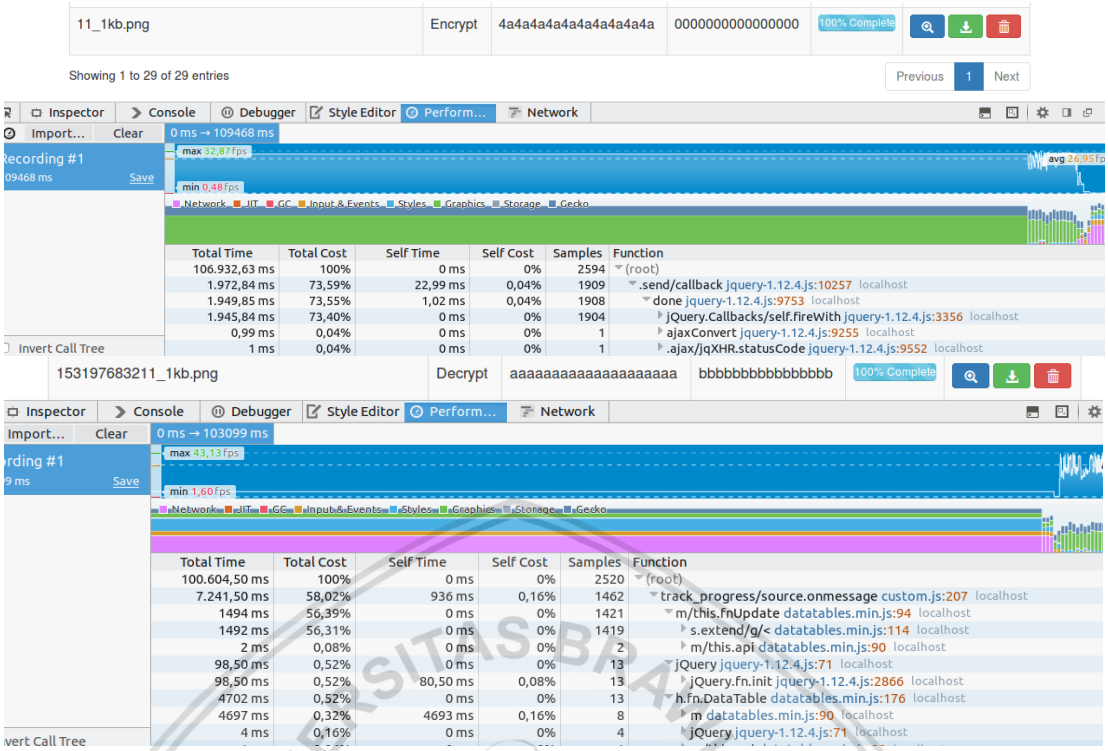
5.5 Pengujian Kinerja Waktu Sistem

5.5.1 Tujuan Pengujian

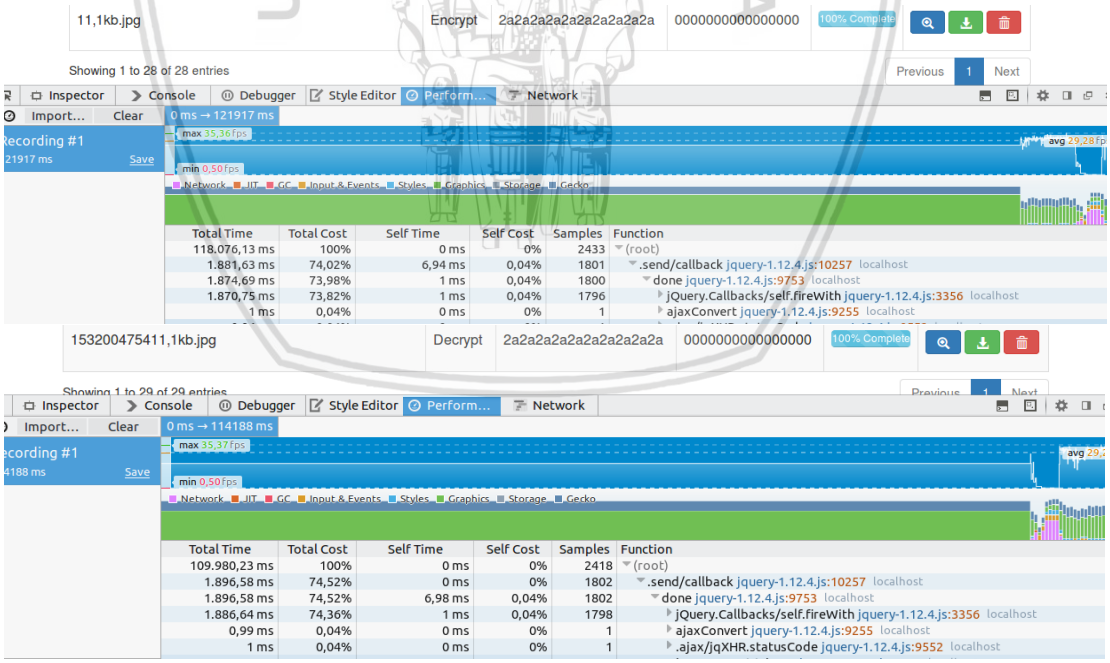
Tujuan pengujian ini ialah untuk mengetahui kinerja dari sistem dengan melakukan pengamatan secara statistik dimana yang akan diamati adalah waktu dari proses yang dijalankan oleh sistem dan juga untuk waktu proses pengiriman data.

5.5.2 Prosedur Pengujian

Untuk melakuan pengujian performance sistem ini, diperlukan bantuan dari sebuah tools dimana tools yang digunakan yaitu web develeper tools yang nantinya langsung dijalankan dari browser dengan menggunakan tools network monitoring sebagai tools untuk melihat proses pengiriman data dan tools perform untuk melihat waktu tempuh berdasarkan proses pengiriman data yang telah selesai dilakukan.

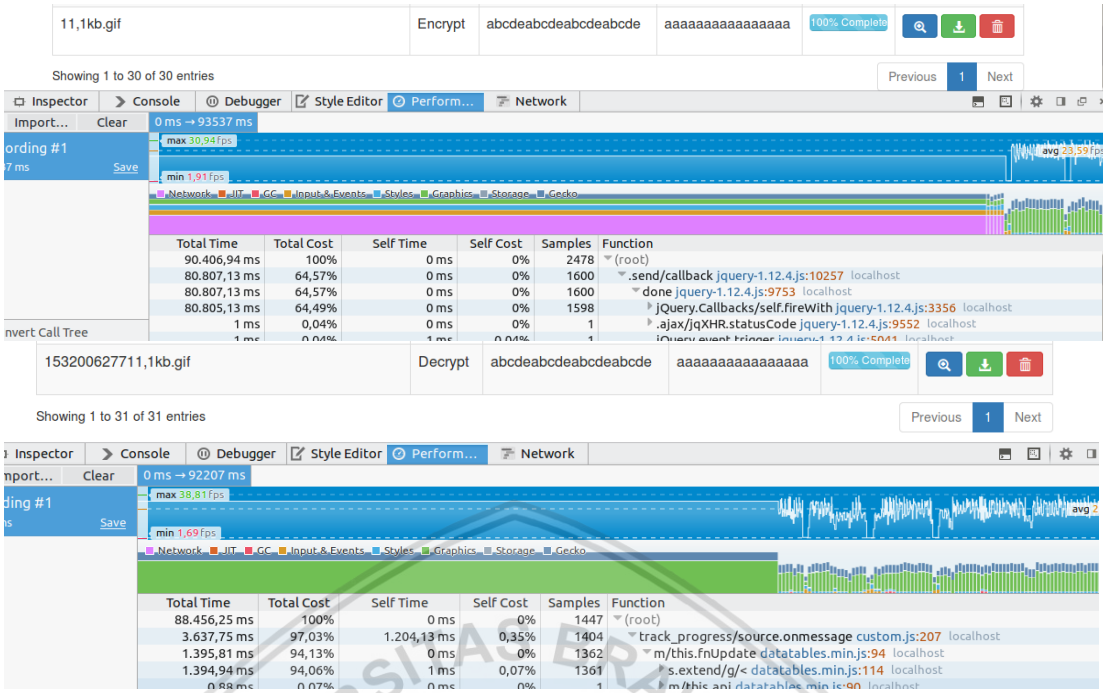


Gambar 5.11 Pengujian Kinerja Waktu Sample PNG

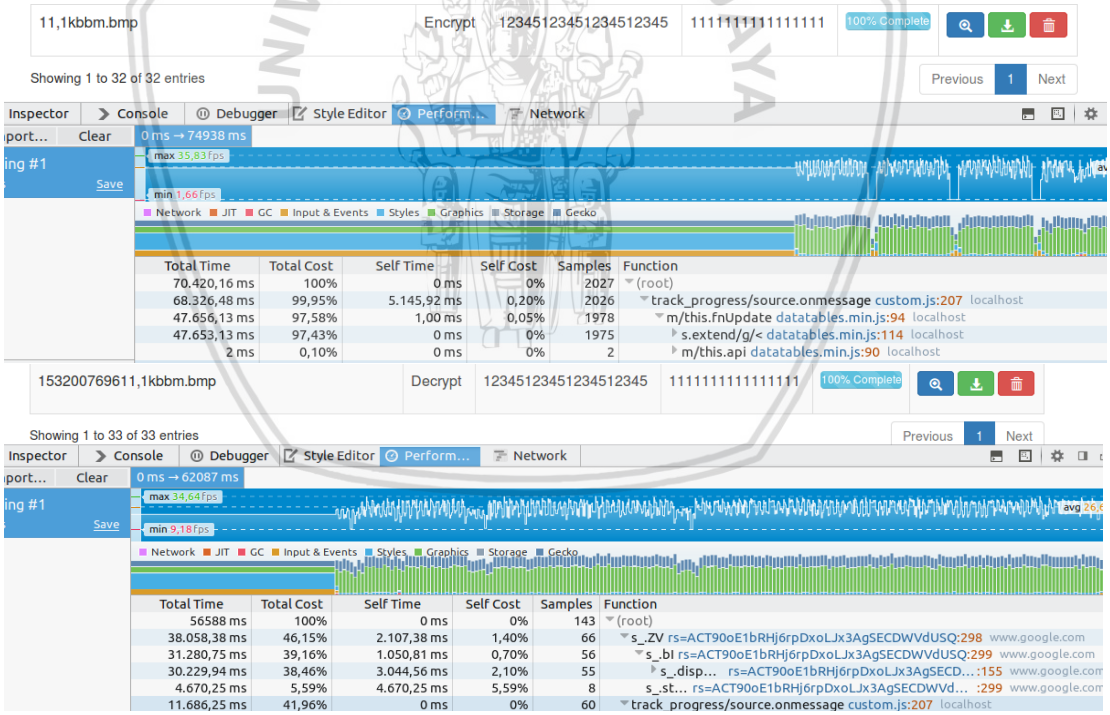


sample2.png	Encrypt	12345123451234512345	0000000000000000	100% Complete	
1527729883sample2.png	Decrypt	12345123451234512345	0000000000000000	100% Complete	

Gambar 5.12 Pengujian Kinerja Waktu Sample JPG



Gambar 5.13 Pengujian Kinerja Waktu Sample GIF



Gambar 5.14 Pengujian Kinerja Waktu Sample BMP

5.5.3 Hasil Pengujian

Akan dilakukan pengujian sebanyak 30 kali dengan skenario yang sama. Menurut Hair et al. (2010) yang menyatakan bahwa ketika *sample* berjumlah 30 atau lebih sedikit jika terdapat nilai normalitas pada *sample* tersebut maka nilai normalitas tersebut sangat mempengaruhi hasil akhir pengujian.

Tabel 5. 11 Kinerja Waktu Pemrosesan

No	Ekstensi	Size	Resolusi (pixels)	Enkripsi	Dekripsi
1	PNG	112 bytes	35 x 12	13,81 detik	11,26 detik
2	PNG	71,1 kb	300 x 168	1,78 menit	1,69 menit
3	PNG	77,8 kb	1299 x 387	1,89 menit	1,74 menit
4	PNG	78,3 kb	1291 x 347	1,93 menit	1,79 menit
5	PNG	78,4 kb	1295 x 321	1,97 menit	1,83 menit
6	PNG	79,8 kb	1273 x 388	2,10 menit	1,98 menit
7	PNG	84,5 kb	1239 x 390	2,52 menit	2,31 menit
8	PNG	86,1 kb	1169 x 396	2,58 menit	2,38 menit
9	PNG	87,0 kb	1255 x 398	2,64 menit	2,44 menit
10	PNG	87,9 kb	1195 x 392	2,68 menit	2,47 menit
11	PNG	89,3 kb	1271 x 424	2,82 menit	2,52 menit
12	PNG	90,3 kb	1232 x 404	2,85 menit	2,65 menit
13	PNG	91,2 kb	1282 x 344	2,91 menit	2,69 menit
14	PNG	94,1 kb	1295 x 363	3,01 menit	2,83 menit
15	PNG	96,2 kb	1122 x 641	3,20 menit	3,09 menit
16	PNG	104,1 kb	720 x 586	3,94 menit	3,54 menit
17	PNG	104,3 kb	272 x 170	3,97 menit	3,55 menit
18	PNG	113,6 kb	841 x 340	4,02 menit	3,89 menit
19	PNG	125,2 kb	1202 x 393	4,94 menit	4,10 menit
20	PNG	130,7 kb	723 x 490	5,03 menit	4,52 menit
21	PNG	162,4 kb	1121 x 397	5,74 menit	5,21 menit
22	PNG	191,5 kb	1366 x 768	6,12 menit	5,97 menit
23	PNG	207,1 kb	400 x 250	6,52 menit	6,31 menit
24	PNG	234,2 kb	589 x 340	7,01 menit	6,82 menit
25	PNG	242,4 kb	601 x 340	7,43 menit	6,94 menit
26	PNG	293,9 kb	743 x 340	7,95 menit	7,23 menit
27	PNG	344,8 kb	668 x 340	8,42 menit	7,87 menit
28	PNG	385,5 kb	523 x 340	8,88 menit	8,05 menit

Tabel 5. 11 Kinerja Waktu Pemrosesan (Lanjutan)

No.	Ekstensi	Size	Resolusi (pixels)	Enkripsi	Dekripsi
29	PNG	503,1 kb	800 x 500	9,31 menit	8,82 menit
30	PNG	1,1 mb	912 x 513	12,2 menit	10,11 menit
31	JPG	112 bytes	35 x 12	15,2 detik	12,9 detik
32	JPG	71, kb	300 x 168	1,96 menit	1,83 menit
33	JPG	77,8 kb	1299 x 387	2,08 menit	1,84 menit
34	JPG	78,3 kb	1291 x 347	2,19 menit	1,89 menit
35	JPG	78,4 kb	1295 x 321	2,22 menit	1,93 menit
36	JPG	79,8 kb	1273 x 388	2,30 menit	2,08 menit
37	JPG	84,5 kb	1239 x 390	2,61 menit	2,41 menit
38	JPG	86,1 kb	1169 x 396	2,68 menit	2,48 menit
39	JPG	87,0 kb	1255 x 398	2,74 menit	2,54 menit
40	JPG	87,9 kb	1195 x 392	2,78 menit	2,57 menit
41	JPG	89,3 kb	1271 x 424	2,92 menit	2,62 menit
42	JPG	90,3 kb	1232 x 404	2,95 menit	2,75 menit
43	JPG	91,2 kb	1282 x 344	3,01 menit	2,79 menit
44	JPG	94,1 kb	1295 x 363	3,11 menit	2,93 menit
45	JPG	96,2 kb	1122 x 641	3,30 menit	3,19 menit
46	JPG	104,1 kb	720 x 586	4,04 menit	3,64 menit
47	JPG	104,3 kb	272 x 170	4,07 menit	3,65 menit
48	JPG	113,6 kb	841 x 340	4,22 menit	3,99 menit
49	JPG	125,2 kb	1202 x 393	4,96 menit	4,20 menit
50	JPG	130,7 kb	723 x 490	5,33 menit	4,62 menit
51	JPG	162,4 kb	1121 x 397	5,94 menit	5,31 menit
52	JPG	191,5 kb	1366 x 768	6,52 menit	6,07 menit
53	JPG	207,1 kb	400 x 250	6,92 menit	6,41 menit
54	JPG	234,2 kb	589 x 340	7,21 menit	6,92 menit
55	JPG	242,4 kb	601 x 340	7,63 menit	7,04 menit
56	JPG	293,9 kb	743 x 340	8,05 menit	7,33 menit
57	JPG	344,8 kb	668 x 340	8,82 menit	7,97 menit
58	JPG	385,5 kb	523 x 340	9,18 menit	8,15 menit
59	JPG	503,1 kb	800 x 500	10,93 menit	8,92 menit
60	JPG	1,1 mb	912 x 513	13,12 menit	10,21 menit

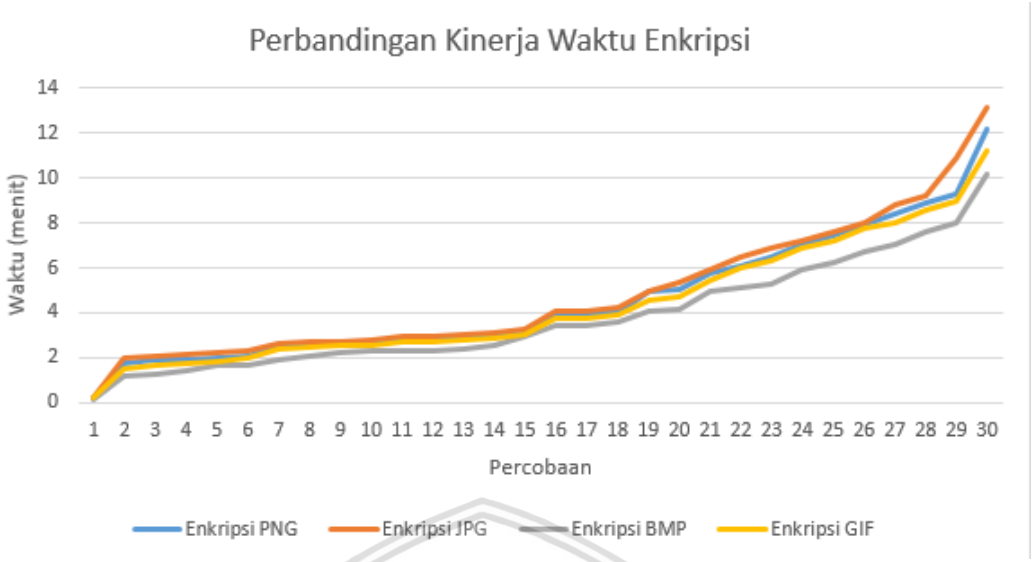
Tabel 5. 11 Kinerja Waktu Pemrosesan (Lanjutan)

No.	Ekstensi	Size	Resolusi (pixels)	Enkripsi	Dekripsi
61	BMP	112 bytes	35 x 12	10,5 detik	11,64 detik
62	BMP	71, kb	168 x 300	1,17 menit	56,58 detik
63	BMP	77,8 kb	1299 x 387	1,29 menit	1,54 menit
64	BMP	78,3 kb	1291 x 347	1,43 menit	1,59 menit
65	BMP	78,4 kb	1295 x 321	1,67 menit	1,63 menit
66	BMP	79,8 kb	1273 x 388	1,69 menit	1,78 menit
67	BMP	84,5 kb	1239 x 390	1,94 menit	2,01 menit
68	BMP	86,1 kb	1169 x 396	2,08 menit	2,18 menit
69	BMP	87,0 kb	1255 x 398	2,24 menit	2,24 menit
70	BMP	87,9 kb	1195 x 392	2,28 menit	2,17 menit
71	BMP	89,3 kb	1271 x 424	2,32 menit	2,32 menit
72	BMP	90,3 kb	1232 x 404	2,35 menit	2,45 menit
73	BMP	91,2 kb	1282 x 344	2,38 menit	2,49 menit
74	BMP	94,1 kb	1295 x 363	2,59 menit	2,53 menit
75	BMP	96,2 kb	1122 x 641	2,92 menit	2,79 menit
76	BMP	104,1 kb	720 x 586	3,44 menit	3,24 menit
77	BMP	104,3 kb	272 x 170	3,47 menit	3,35 menit
78	BMP	113,6 kb	841 x 340	3,62 menit	3,59 menit
79	BMP	125,2 kb	1202 x 393	4,04 menit	3,91 menit
80	BMP	130,7 kb	723 x 490	4,17 menit	4,12 menit
81	BMP	162,4 kb	1121 x 397	4,94 menit	5,01 menit
82	BMP	191,5 kb	1366 x 768	5,11 menit	5,67 menit
83	BMP	207,1 kb	400 x 250	5,32 menit	6,01 menit
84	BMP	234,2 kb	589 x 340	5,91 menit	6,42 menit
85	BMP	242,4 kb	601 x 340	6,23 menit	6,54 menit
86	BMP	293,9 kb	743 x 340	6,75 menit	7,03 menit
87	BMP	344,8 kb	668 x 340	7,02 menit	7,37 menit
88	BMP	385,5 kb	523 x 340	7,58 menit	7,65 menit
89	BMP	503,1 kb	800 x 500	8,01 menit	8,32 menit
90	BMP	1,1 mb	912 x 513	10,2 menit	9,55 menit
91	GIF	112 bytes	35 x 12	12,25 detik	10,96 detik
92	GIF	71, kb	168 x 300	1,50 menit	1,47 menit

Tabel 5. 11 Kinerja Waktu Pemrosesan (Lanjutan)

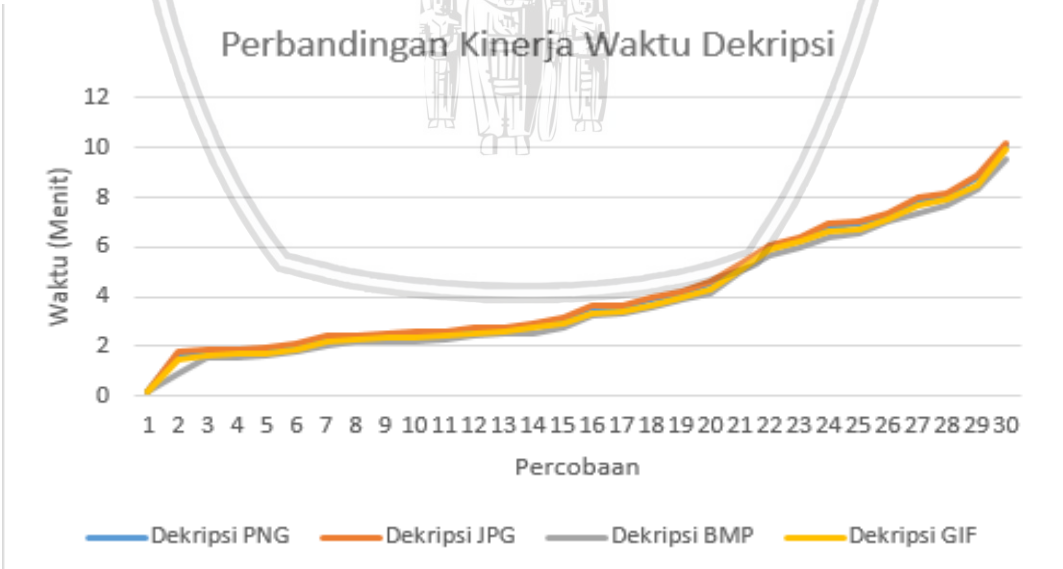
No.	Ekstensi	Size	Resolusi (pixels)	Enkripsi	Dekripsi
93	GIF	77,8 kb	1299 x 387	1,69 menit	1,64 menit
94	GIF	78,3 kb	1291 x 347	1,73 menit	1,69 menit
95	GIF	78,4 kb	1295 x 321	1,87 menit	1,73 menit
96	GIF	79,8 kb	1273 x 388	2,01 menit	1,88 menit
97	GIF	84,5 kb	1239 x 390	2,42 menit	2,21 menit
98	GIF	86,1 kb	1169 x 396	2,48 menit	2,28 menit
99	GIF	87,0 kb	1255 x 398	2,54 menit	2,34 menit
100	GIF	87,9 kb	1195 x 392	2,58 menit	2,37 menit
101	GIF	89,3 kb	1271 x 424	2,72 menit	2,42 menit
102	GIF	90,3 kb	1232 x 404	2,75 menit	2,55 menit
103	GIF	91,2 kb	1282 x 344	2,81 menit	2,59 menit
104	GIF	94,1 kb	1295 x 363	2,91 menit	2,73 menit
105	GIF	96,2 kb	1122 x 641	3,02 menit	2,89 menit
106	GIF	104,1 kb	720 x 586	3,74 menit	3,34 menit
107	GIF	104,3 kb	272 x 170	3,77 menit	3,45 menit
108	GIF	113,6 kb	841 x 340	3,92 menit	3,69 menit
109	GIF	125,2 kb	1202 x 393	4,54 menit	4,01 menit
110	GIF	130,7 kb	723 x 490	4,73 menit	4,32 menit
111	GIF	162,4 kb	1121 x 397	5,44 menit	5,11 menit
112	GIF	191,5 kb	1366 x 768	6,01 menit	5,87 menit
113	GIF	207,1 kb	400 x 250	6,32 menit	6,21 menit
114	GIF	234,2 kb	589 x 340	6,91 menit	6,62 menit
115	GIF	242,4 kb	601 x 340	7,23 menit	6,74 menit
116	GIF	293,9 kb	743 x 340	7,75 menit	7,13 menit
117	GIF	344,8 kb	668 x 340	8,02 menit	7,67 menit
118	GIF	385,5 kb	523 x 340	8,58 menit	7,95 menit
119	GIF	503,1 kb	800 x 500	9,01 menit	8,52 menit
120	GIF	1,1 mb	912 x 513	11,2 menit	9,95 menit

Pada Tabel 5. 11 didapatkan hasil waktu pemrosesan enkripsi dan dekripsi pada sistem dimana keempat file ekstensi yang diuji.



Gambar 5.15 Perbandingan Kinerja Waktu Enkripsi

Pada Gambar 5. 15 didapatkan hasil waktu pemrosesan enkripsi pada sistem dimana keempat file ekstensi yang diuji. Berdasarkan grafik, diperoleh hasil rata-rata waktu terbesar pada file ekstensi JPG sedangkan untuk hasil rata-rata waktu terkecil pada file ekstensi BMP. Hal tersebut dikarenakan pada ekstensi file tersebut walaupun memiliki ukuran file dan resolusi yang sama dengan file ekstensi lain secara kualitas ekstensi JPG masih lebih baik dibandinkan dengan ekstensi BMP.



Gambar 5.16 Perbandingan Kinerja Waktu Dekripsi

Pada Gambar 5. 16 Berdasarkan grafik, didapatkan hasil waktu pemrosesan dekripsi dengan rata-rata waktu terkecil pada file ekstensi BMP sedangkan untuk hasil rata-rata waktu terbesar pada file ekstensi JPG. Hal tersebut dikarenakan pada ekstensi file tersebut walaupun memiliki ukuran file dan resolusi yang sama

dengan file ekstensi lain secara kualitas ekstensi JPG masih lebih baik dibandinkan dengan ekstensi BMP.

5.6 Pengujian Keamanan

5.6.1 Tujuan Pengujian

Tujuan pengujian ini ialah untuk mengetahui apakah data dapat terbaca oleh pengguna lain baik sebelum maupun setelah dilakukan enkripsi.

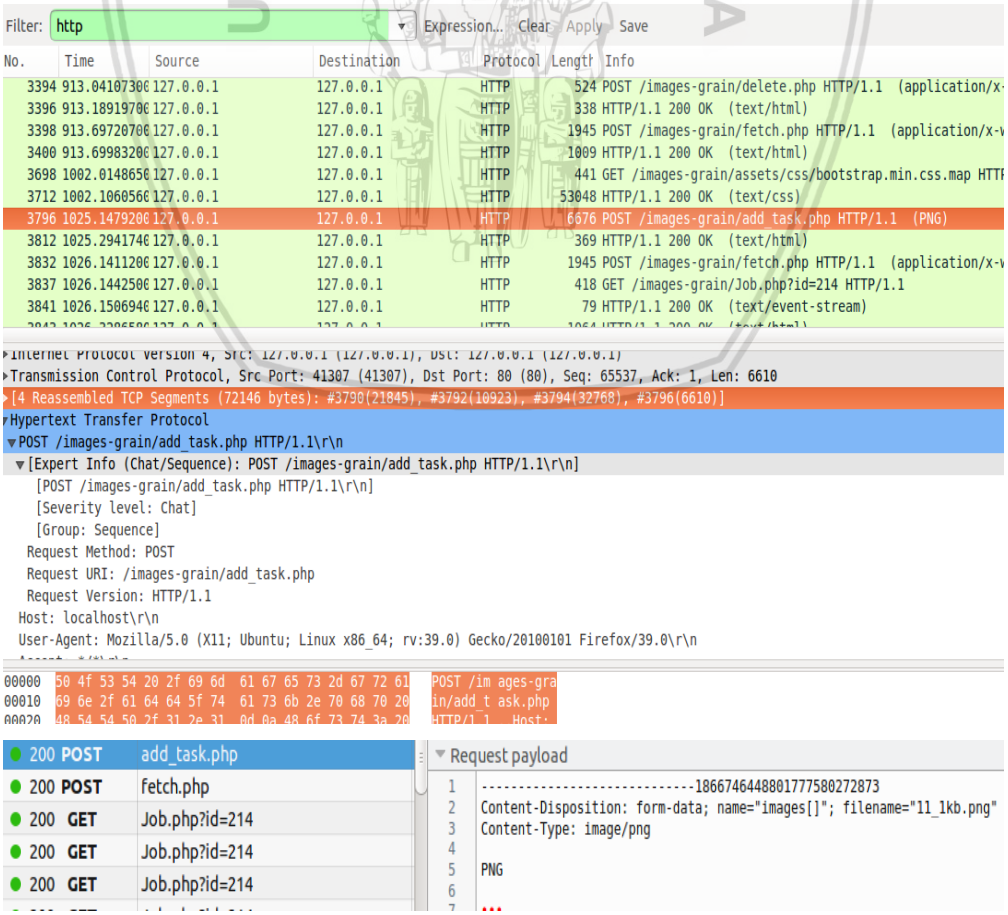
5.6.2 Prosedur Pengujian

Untuk melakuan pengujian enkripsi dekripsi ini, diperlukan sebuah tools yang dapat melakukan trace terhadap file yaitu wireshark dimana mampu menangkap paket data atau informasi yang telah melewati jaringan nantinya interface yang digunakan yaitu lo > localhost karena penelitian ini masih menggunkan localhost sebagai local server untuk dapat menghosting file pada web aplikasi dan protokol http digunakan untuk dapat menampilkan data atau informasi didalam proses berjalan nya enkripsi.

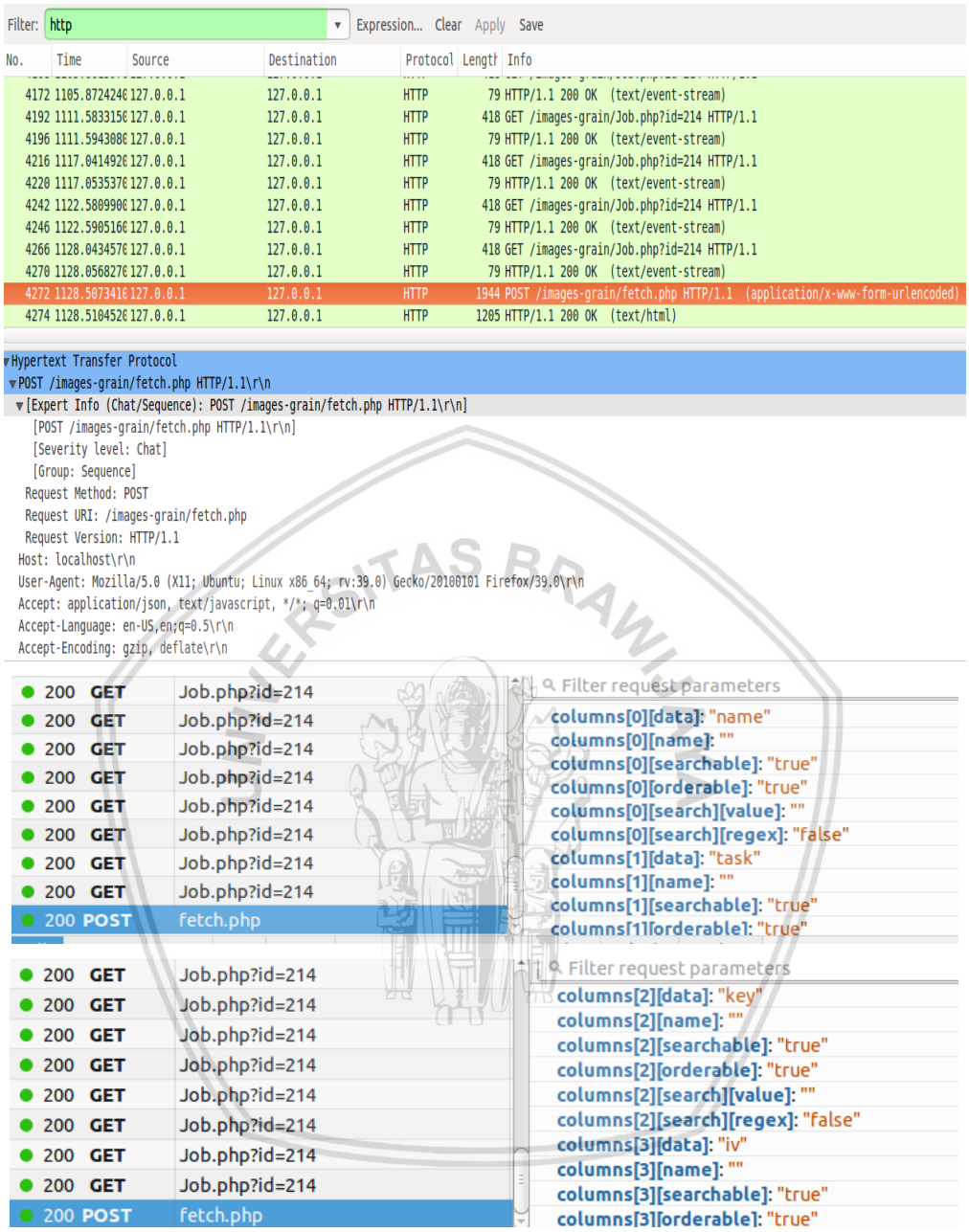
5.6.3 Hasil Pengujian

Berikut merupakan hasil pengujian keamanan

Sebelum Proses Enkripsi



Sebelum Proses Enkripsi

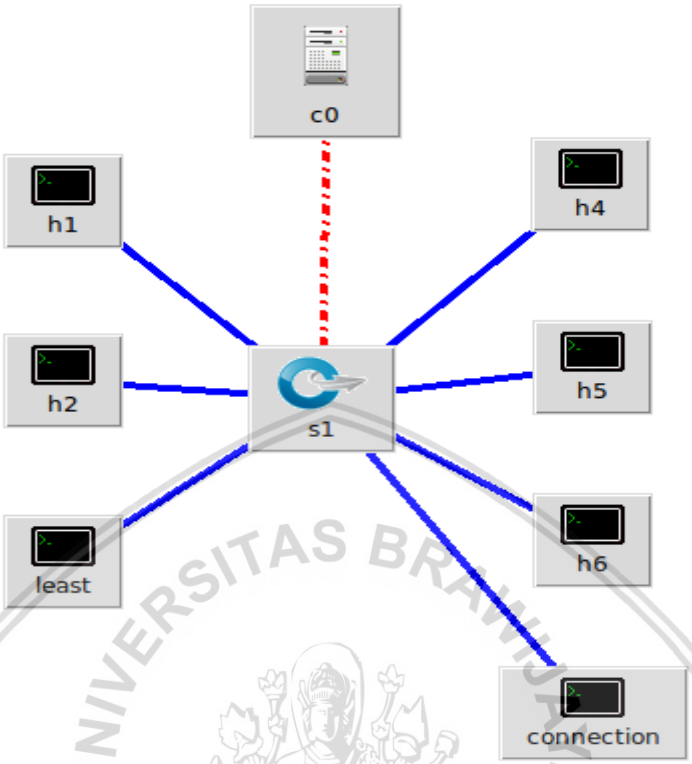


Gambar 5.17 Hasil Pengujian Keamanan

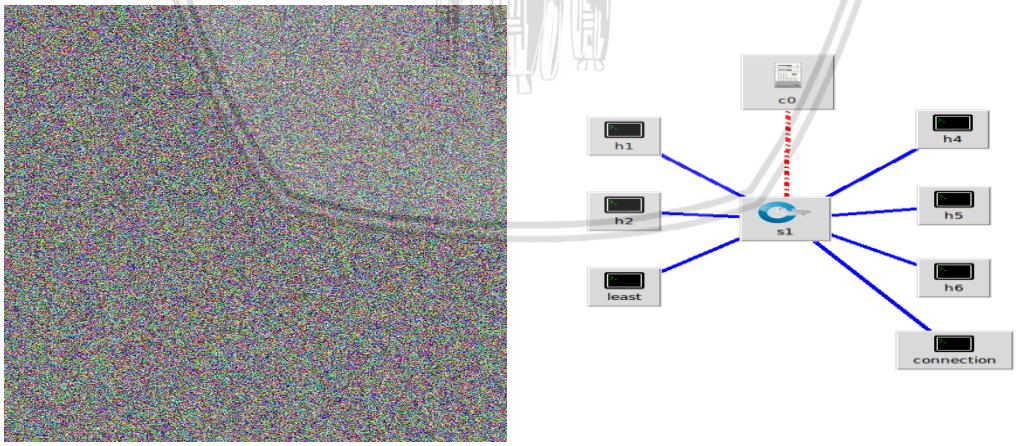
Pembahasan :

Digunakan method post untuk dapat melihat data baik sebelum proses enkripsi maupun setelah proses enkripsi. Sebelum proses enkripsi data sensitif masih terlihat namun ketika proses enkripsi dijalankan data tersebut tidak terbaca yaitu hanya menampilkan parameternya saja tidak dengan isinya.

Sample2



Gambar 5.18 Gambar Asli Sample1

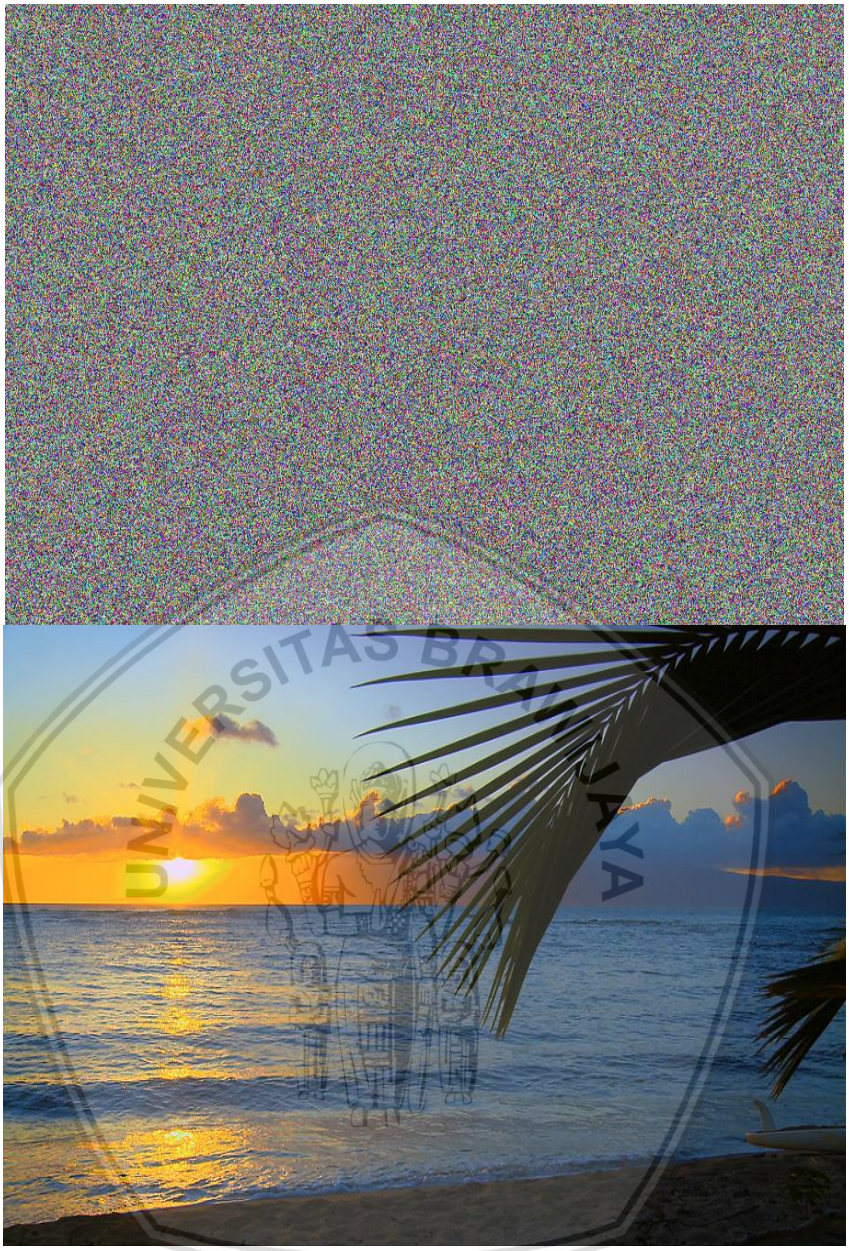


Gambar 5.19 Hasil Enkripsi dan Dekripsi

Data dengan size tinggi

Sample4

Gambar 5.20 Gambar Asli Sample2



Gambar 5.21 Hasil Enkripsi dan Dekripsi

Tabel 5. 12 Hasil Enkripsi dan Dekripsi

Gambar	Size Awal	Size Enkripsi	Size Dekripsi	Resolusi
Gambar 5.19	11,2 kb	674 kb	11,2 kb	430 x 534
Gambar 5.21	1,01 MB	1,34 MB	820 kb	912 x 513

Pada kedua sample hasil akhirnya adalah Gambar tidak mengalami perubahan baik dari segi warna atau resolusi setelah di enkripsi dan dekripsi

namun terdapat perbedaan diantara kedua sample dimana pada sample2 ukuran file tetap sedangkan pada sample4 ukuran file berubah setelah dilakukan proses dekripsi hal ini dikarenakan kombinasi warna pada sample2 tidak terlalu kompleks dibandingkan dengan kombinasi warna pada sample4 tetapi tidak mengurangi kualitas dari Gambar tersebut hasilnya akan tetap terlihat sama baik pada saat sebelum maupun sesudah dilakukan proses enkripsi dan dekripsi

BAB 6 PENUTUP

6.1 Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan, dapat disimpulkan beberapa hal mengenai algoritme Grain.

1. Algoritme Grain dapat diimplementasikan untuk dapat melakukan enkripsi pada format gambar di mana dalam melakukan enkripsi gambar ini digunakan konsep bit per pixel, bit diambil dari komponen gambar yaitu RGB (red,green,blue) yang terdiri dari Merah, hijau dan biru dimana masing-masing menggunakan 8 bit yang memiliki nilai integer dari 0 hingga 255. Ini membuat 256 * 256 * 256 = 16777216 warna yang memungkinkan. Setelah set up untuk mendapatkan nilai bit dari RGB telah dilakukan kemudian proses enkripsi dijalankan.
2. Hasil pembentukan keystream pada sistem sesuai dengan test vector yang terdapat pada paper algoritme Grain V1.
3. Algoritme Grain yang diterapkan pada enkripsi gambar memiliki perbandingan waktu pemrosesan enkripsi dan dekripsi yang bervariasi. Berdasarkan hasil pengujian, didapatkan waktu pemrosesan enkripsi dan dekripsi dengan rata-rata waktu terkecil pada file ekstensi BMP sedangkan untuk hasil rata-rata waktu terbesar pada file ekstensi JPG. Hal tersebut dikarenakan pada ekstensi file tersebut walaupun memiliki ukuran file dan resolusi yang sama dengan file ekstensi lain secara kualitas ekstensi JPG masih lebih baik dibandinkan dengan ekstensi BMP dan juga ekstensi file lainnya.

6.2 Saran

Saran pada penelitian ini untuk dapat dikembangan pada penelitian selanjutnya yaitu mengembangkan aplikasi yang terdapat pada sistem dari aspek perangkat lunaknya agar nantinya sistem tersebut dapat memenuhi standar aplikasi yang dibutuhkan baik fungsional maupun non-fungsional sesuai kebutuhan user. Diharapkan penelitian ini dapat digunakan sebagai rujukan untuk pengembangan selanjutnya.

DAFTAR PUSTAKA

Ade, 2009. *Image Processing.* [Online] Available at: http:ndoware.com/image-processing.html [Diakses 20 January 2018].

Chandrasekaran , J. & Jayaraman, D. S., 2015. A Fast and Secure Image Encryption Algorithm Using Number Theoretic Transforms and Discrete Logarithms. *Institute of Electrical and Electronics Engineers,* pp. -.

Hair, J.F., Black, W.C., Babin. B. J. & Andreson R. E. 2010. Multivariate Data Analysis. Edisi 7. Upper Saddle River: Prentice Hall.

Hell, M., Johansson, T. & Meier, W., 2006. Grain - A Stream Cipher for Constrained Environments.

Herlambang, S., 2010. Studi dan Analisis Grain Cipher.

Kromodimoeljo, S., 2010 . *Teori dan Aplikasi Kriptografi.* s.l.: SPK IT Consulting.

Kumar, S. & Srivastava, S., 2014. Image Encryption using Simplified Data Encryption Standard (S-DES). *International Journal of Computer Applications,* p. 104.

Loukhaoukha, K., Chouinard, J.-Y. & Berdai, A., 2012. A Secure Image Encryption Algorithm Based on Rubik's Cube Principle. *Journal of Electrical and Computer Engineering,* Volume Volume 2012 (2012), p. 13.

Padate, R. & Patel, A., 2015. Image Encryption And Decryption Using AES Algorithm. *International Journal Of Electronics And Communication Engineering & Technology,* pp. 23-29.